6. WEB-ДИЗАЙН

Усі Web-сторінки створюються на основі мови HTML (*HyperText Markup Language – мова розмітки гіпертексту*).

Вона була створена вченими Європейського центру ядерних досліджень (CERN, м. Женева). Наприкінці 80-х років у CERN зайнялись проблемою збереження і відображення даних, які отримували колеги-фізики. Складність полягала у тому, що кожний фахівець, який приїздив до Центру, застосовував власні методи відображення інформації, і потрібно було терміново створити універсальну систему, яка б не залежала від комп'ютерної платформи і була б досить простою.

Ідея розв'язання проблеми обміну документами між різними комп'ютерами полягала у тому, що документи мали бути розмічені за допомогою визначеного коду – HTML. Такі документи могли б читатися на будь-якому комп'ютері, на якому встановлена одна програма перегляду – браузер.

HTML являє собою набір команд, що описують структуру документа. Така мова розмітки дозволяє виділити в документі окремі логічні частини (заголовки, абзаци, списки і т.д.), але не задає конкретні атрибути форматування. Конкретний вид форматування задає використовуваний при читанні документа браузер, що забезпечує відображення інформації на моніторі.

Конструкції цієї мови – *теги* – дозволяють керувати шрифтом, кольором тексту і фону, визначати посилання, вставляти графіку, аудіо і відео. Теги ніколи не відображаються при перегляді сторінки – вони служать для управління оформленням. Якщо ж при відкритті сторінки ви бачите теги, це означає, що при написанні коду були допущені помилки.

Гіпертекст – текст, у якому містяться посилання на інші текстові документи. Це дає можливість при читанні тексту легко і швидко переходити до іншої зв'язаної з ним текстової інформації. Зв'язаний з посиланням текст може бути фрагментом того ж самого документа, чи іншим текстовим документом, що зберігається на будь-якому іншому ПК в глобальній мережі.

Документи, підготовлені мовою HTML, називають HTML-документами (Web-сторінка, HTML-сторінка, Internet-сторінка). Для того, щоб переглянути HTML-код сторінки, потрібно після завантаження її в браузері вибрати команду *Вид – Просмотр HTML-кода*. Код поточної сторінки відобразиться у вікні Блокнота. Віднедавна, деякі автори сайтів, почали захищати свою власність і блокувати копіювання змісту і відповідно перегляд коду.

Web-сайт – сукупність Web-сторінок, об'єднаних однією загальною темою, поміщених, як правило, на одному вузловому комп'ютері і контролюються однією людиною чи групою людей.

Посилання в гіперпертекстових документах називаться *гіперпосиланнями (лінк, гіперлінк)*. В якості гіперпосилання може бути використаний будь-який текст, символ чи рисунок.

Однією з цілей проекту World Wide Web була розробка стандартного способу вказівки посилань на доступні Internet ресурси. Для рішення цієї задачі було введене поняття URL.

URL (Uniform Resource Locator - універсальний покажчик ресурсу) являє собою адресу ресурсу в Internet.

Приклад URL: http://www.mysite.ua/my-page.htm

6.1. Проектування структури сайту

Приступаючи до розробки сайту потрібно чітко визначити його призначення та аудиторію на яку він розрахований. Єдиної класифікації сайтів не існує, але є багато різних підходів до визначення цього питання. Поклавши в основу класифікації призначення сайтів, можна умовно розбити їх на наступні групи:

1) **Комерційні сайти.** Основна мета будь-якого комерційного сайту полягає в обслуговуванні користувачів таким способом, що приносить компанії пряму або побічну вигоду. Комерціалізація Internet з кожним роком стає все інтенсивнішою. З'явився вид бізнесу, який називають електронним, тобто ділова активність, що використовує можливості глобальних інформаційних мереж для отримання прибутку. Можна також виділити певні підгрупи сайтів:

а. Web-представництво фірми – призначені для підтримки бізнесу якої-небудь фірми чи організації, та використовуються для поширення інформації про продукти і послуги, що надаються фірмою, і способах зв'язку з фірмою;
б. Електронна крамниця – продаж товарів через мережу;
в. Електронний аукціон – аналог звичайного аукціону;
г. Електронна біржа – торги великими партіями для юридичних осіб;
д. Платне надання інформації та програмних продуктів (реферати, аналітична інформація, оновлення програм і т.п.).

2) **Інформаційні сайти.** Ставлять перед собою задачу інформування користувачів у певних сферах громадського життя. Як правило, інформаційні сайти відрізняються великим обсягом розміщених на них матеріалів (текстових, табличних і графічних), їх можна умовно розбити на:

е. урядові;
ж. освітні;
з. новин; некомерційних фірм і організацій;
и. релігійних груп;
к. суспільні.

3) **Розважальні сайти.** Мають на меті розважити своїх відвідувачів. Як правило, запропонованим товаром саме і є розвага. Розважальні сайти, в основному, насичені графікою, анімацією і спеціальними ефектами. Для повноцінної роботи потрібні потужні канали зв'язку для швидкого завантаження і адекватного відображення інформації.

4) **Навігаційні сайти.** Допомагають користувачам у пошуку потрібної інформації через Internet, їх часто називають Internet-порталами. До них можна

віднести пошукові системи, каталоги, рейтинги і деякі довідкові системи.

5) **Художні сайти.** Є самовираженням автора, найчастіше, Web-дизайнера. Вплив користувача на художній сайт зводиться до того, що він може подумки схвалити його або ні. Досить часто сайти виконані в авангардному дизайні.

6) **Персональний сайт.** Часто називається персональною домашньою сторінкою. На таких сайтах розповідається про автора, коло його захоплень та розміщується інформація, якою він бажає поділитись з аудиторією. Практично всі відомі люди, на даний момент мають персональні сторінки (наприклад – Мороз Олександр Олександрович www.moroz.com.ua). Як правило, сайти цього типу не мають на меті отримання прибутку.

7) **Комбіновані сайти.** Поєднують у собі два або більш типів сайтів і їх переваги (комерційний Internet-портал, інформаційно-комерційний сайт і т.п.).

Якщо розглядати сайти для бізнесу, то доцільність створення того чи іншого типу сайту визначається наступними основними критеріями:

1) *Коло розв'язуваних сайтом задач*. Необхідно вибрати основну мету створення сайту і другорядні цілі. Від цього багато в чому будуть залежати загальна ідеологія сайту, його структура, компонування, система навігації і т.д.

Можливі цілі створення сайту:

- повна і оперативна інформація про діяльність компанії, доступна в будь-який час з будь-якої точки світу;
- формування у споживачів, партнерів та інвесторів сприятливого іміджу сучасної компанії;
- інформаційна підтримка дилерів і партнерів компанії;
- продаж товарів/послуг;
- проведення тендерів на постачання сировини, матеріалів і устаткування;
- вихід на ринки СНД і Далекого Зарубіжжя, що неможливо належною мірою охопити стандартними методами реклами;
- підтримка теле/радіо/друкованої рекламної кампанії. Організація рекламної кампанії з посиланням на сайт, як на джерело найбільш повної і оперативної інформації про предмет реклами і т.д.

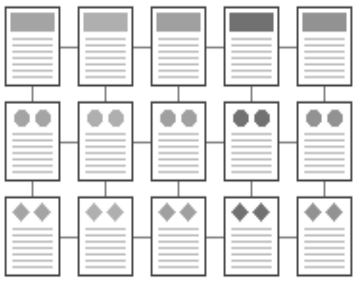
2) *Бюджет сайту*. У залежності від нього буде залежати вибір форми документа, на основі якого буде розроблятися сайт. Від бюджету прямо залежить також і загальний обсяг сайту, рівень його технічної складності, наявність додаткових сервісів, строки створення, число майбутніх відвідувачів, а отже, той прибуток, що зможе приносити сайт його власникові. Фактично в багатьох випадках від бюджету залежить якість даного сайту і те наскільки серйозно його будуть сприймати відвідувачі. При плануванні бюджету потрібно враховувати як витрати на розробку сайту, так і витрати на його постійний супровід (своєчасне оновлення та поповнення).

3) *Інформаційне наповнення сайту*. Підбір матеріалу надзвичайно важливий фактор. Потрібно оцінити якість матеріалу, його цікавість для відвідувачів. Відібраний матеріал групують по темам, які і будуть визначати розділи майбутнього сайту. Якщо матеріалу по вибраній темі набирається досить багато, слід відсортувати його по ступені важливості.

Матеріали, відібрані для сайту, потрібно організувати у визначену структуру (табл. 6.1).

Таблиця 6.1

Типи структур сайту

Схема	Опис
Лінійна структура 	Сторінки сайту розташовуються строго одна за одною. Зручна для створення невеликого по кількості сторінок сайту з малою кількістю гіперпосилань і послідовним викладом матеріалів.
Структура у виді ґрат 	Ґрунтується на побудові системи навігації сайту, коли між вертикальними і горизонтальними елементами (сторінками) є взаємний зв'язок і можливість швидкого переходу з однієї сторінки на іншу, без необхідності відвідування проміжних сторінок. Подібна структура приводить до зайвого збільшення гіперпосилань і застосування її обмежене для об’ємних сайтів.
Оптимальна структура (деревоподібна) 	На верхньому рівні знаходиться початкова сторінка сайту, з якої відвідувач за допомогою меню чи посилань може переходити на сторінки наступного рівня. Ця структура може складатися з декількох рівнів підпорядкування. Дуже важливо визначити оптимальне співвідношення між кількістю рівнів (глибиною коренів чи висотою дерева) і кількістю варіантів вибору на кожному рівні (шириною дерева).
Різновид деревоподібної структури 	Використовується в особливо складних сайтах, що мають декілька сот окремих html-сторінок. Принципово відрізняється тим, що має велику глибину сайту. При трирівневій системі навігації можна розмістити приблизно 15 000 тисяч сторінок. Обсяг інформації такого сайту може скласти кілька десятків тисяч сторінок.

Відвідувач може скласти уявлення про структуру сайту за допомогою засобів навігації: списків-посилань, меню і т.д. Для кращої орієнтації відвідувачів до складу сайту може бути включена окрема сторінка, так звана **карта** сайту.

Крім загальної структури потрібно спроектувати зміст окремих його сторінок: їх дизайн, інформаційне наповнення, ілюстративне оформлення, розробити переходи та контекстові посилання на інші сторінки сайту чи матеріали в мережі.

Сайт може бути статичним і динамічним. Статичні це в основному інформаційні сайти. Динамічні, орієнтовані на спілкування з користувачем, і формуються спеціальними програми у відповідь на дані, що він задає.

6.2. HTML-коди

Перш ніж створювати свої власні HTML-сторінки, слід розібратися в їх структурі та принципах функціонування, а також знати код HTML-документа. Мова HTML дозволяє формувати різну гіпертекстову інформацію на основі структурованих документів, а браузер визначає сформовані посилання і, через протокол передачі гіпертексту HTTP, відкриває доступ до документа іншим користувачам Internet.

Для створення Web-сторінок необхідний текстовий редактор, що дозволяє зберігати файли в ASCII коді, а це означає, що HTML-документ не містить яких-небудь знаків форматування тексту. Він може містити тільки літери, цифри, розділові знаки і деякі інші друкарські символи.

Найпростішим редактором, що дозволяє створювати Web-сторінки, є Блокнот, який вбудований в операційну систему Windows. Написавши чергову сторінку в HTML-тегах, потрібно зберегти її з розширенням *.html. Використання спеціальних програм дозволить прискорити час створення Web-сторінки, але не позбавить від необхідності знання HTML-коду.

HTML-документ, по суті, є звичайним текстовим файлом. Редагувати Web-сторінки, опубліковані в мережі Internet, може лише той, хто їх створив, а не будь-який користувач, оскільки кожна сторінка має свою унікальну адресу (URL) та захищена паролем. Існування двох різних сторінок з однаковою адресою виключене. HTML-сторінка містить як звичний текст, так і спеціальні команди розмітки (tags або теги), вкладені в кутові дужки (< і >).

Теги мови HTML задають правила, за якими браузер відображає документ на екрані: розміщення тексту у вікні, представлення графічних об'єктів (малюнків), а також виведення звукових файлів, відеокліпів і т.д.

Кожен тег має однаковий принцип написання:

<ІМ’Я ТЕГА> Вміст тега </ ІМ’Я ТЕГА >

Знаки «більше», «менше» вжито для розділення тегів один від одного і повідомлення браузеру, що це команда, а не звичайний текст.

Комбінація з відкриваючого і закриваючого тега називається контейнером тегів. „Вмістом тега” може бути текст чи інші теги.

Теги не чутливі до регістра. Це означає, що, наприклад, HTML-тег <body> буде сприйнятий браузером так само, як тег <BODY> або <bOdY>.

Теги не чутливі до їх розташування на сторінці. Їх можна розмістити в один рядок, але для зручності їх перегляду при написанні тексту web-сторінки, рекомендується розташовувати теги ступінчасто, щоби теги старшого рівня були лівіше від тегів нижчого рівня.

Будь-який текстовий документ складається з окремих об’єктів: заголовків, абзаців, малюнків, таблиць, рисунків. Ці об’єкти, переведені у формат HTML, зображаються у вигляді елементів HTML. Наприклад елементом є заголовок сторінки: <TITLE> Приклад </TITLE>.

Всі елементи, передбачені в HTML, можна умовно поділити на декілька категорій:

- структурні – елементи, обов'язкові для документа, що відповідає стандарту HTML (наприклад, HTML, HEAD, BODY і TITLE);
- блокові – елементи, призначені для форматування цілих текстових блоків (наприклад, DIV, H1, H2, P), часто відокремлюються від іншої частини документа пропуском рядка;
- текстові – елементи, що задають розмітку шрифту (I, B, U, BIG, SMALL і ін.), розмітку тексту (STRONG, VAR, CITE);
- спеціальні – елементи порожнього рядка (BR, HR), впроваджені елементи (IMG, MAP, OBJECT), якірний елемент (A), елементи таблиці (TABLE), елементи фреймів тощо.

Структура HTML-документа

HTML-документ можна умовно розбити на три частини (табл. 6.2).

Таблиця 6.2

Структура HTML-документа

<!DOCTYPE >	Службова інформація для браузера
<HTML>	
<HEAD> <TITLE> Заголовок вікна браузера </TITLE> </HEAD>	Заголовна частина документа, до якої вміщена назва документа, а також службова інформація для серверів, описи невеликих програм-сценаріїв.
<BODY> Вміст сторінки </BODY>	Тіло документа
</HTML>	

<HTML> – тег використовується для відкриття HTML-документа, з нього починається кожна Web-сторінка і закінчується закриваючим тегом </HTML>.

<HEAD> – будь-який HTML-документ складається як мінімум з двох частин: заголовка і власне документа. Даний тег визначає заголовок Web-сторінки і повинен мати обов'язковий закриваючий тег </HEAD>.

Між тегами <TITLE> і </TITLE> поміщається назва документа. По ній браузери можуть знайти інформацію, тому місце для назви завжди визначене – вгорі і окремо від вмісту HTML-документа. Відображається назва в заголовку вікна браузера, тому на нього накладається обмеження: не більше 40 символів.

<BODY>. Даний тег містить в собі безпосередньо документ, який відображається браузером. Також необхідний закриваючий тег </BODY>.

На рис. 6.1 наведено приклад застосування цих тегів.

Теги можуть бути:

- парними – мають початковий і кінцевий тег (наприклад, <HEAD> і </HEAD>, <BODY> і </BODY>)
- одиночними (наприклад, <BR>, <!DOCTYPE>).

Часто теги, крім оформлення містять додаткові елементи, які називаються *атрибутами*. Наприклад, якщо в тег тіла документа <BODY> ввести додатковий елемент <BODY bgcolor=”yellow”>, то це означатиме, що документ має відображатися на жовтому тлі. Слово bgcolor є атрибутом, а yellow – значенням атрибуту.

```
<HTML>
    <HEAD>
        <T1TLE>
            Проста
         Web-сторінка
        </TITLE>
    </HEAD>
    <BODY>
        Це найпростіша
        Web-сторінка.
    </BODY>
</HTML>
```

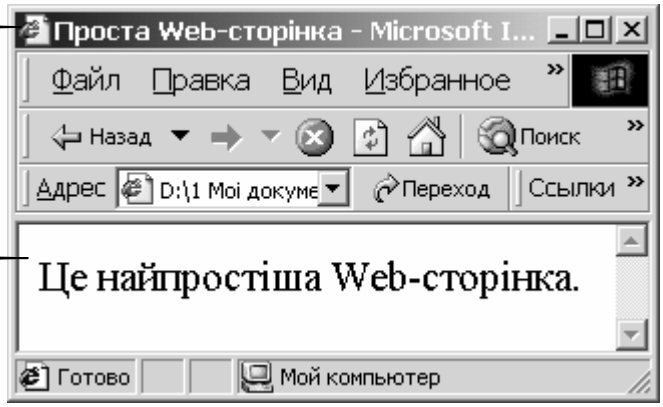


Рис. 6.1. Зображення Web-сторінки в браузері

Після написання початкового коду HTML-документа і перед розміщенням його в мережі Internet, необхідно оцінити результат роботи з точки зору користувача. Саме на цій стадії деколи виникають труднощі, оскільки готовий HTML-документ, тобто Web-сторінка, що відображається в браузері, не знаходиться під безпосереднім контролем автора, як це звично буває у випадку з надрукованими документами.

Відображення залежить від того, який браузер використовує користувач і як він набудований – одна і та ж сторінка може в різних браузерах відображатися по-різному. Також представлення документа залежить від настройок монітора користувача, тобто від встановленої кількості точок на екрані і колірної схеми. Тому для перевірки створеного документа рекомендується використовувати декілька браузерів (як мінімум два).

На сьогоднішній момент найпоширенішими браузерами для сімейства операційних систем Windows, як уже відмічалось, є Internet Explorer корпорації Microsoft, Netscape Navigator компанії Netscape і Opera фірми Opera Software.

Якщо отримане зображення у браузері вимагає коригування, потрібно внести зміни в тексті сторінки, зберегти її і натиснути кнопку „Обновити” на браузері для перевірки. Коригування виконують доки зображення не буде задовольняти повністю.

6.2.1. Основні теги

Всі теги приведені не по порядку їх розміщення в документі і важливості, а згруповані по призначенню в алфавітному порядку.

Теги форматування Web-сторінки

<!-- текст коментарю --> – коментар, який інколи вставляється для пояснень певних блоків програмного коду, читачу в браузері не виводиться.

<ADDRESS>...</ADDRESS> – відомості про автора сторінки, дату створення і оновлення. Виділяється курсивом.

<BASE> – указує повну базову адресу, де зберігаються HTML-документи. Атрибути:

- href ="..." – URL-адреси;
- target="..." – ім'я кадру.

<BODY>...</BODY> – контейнер, в який поміщується текст (тіло) Web-сторінки. Є обов'язковим елементом HTML-документа.

<!DOCTYPE > – містить інформацію про версію мови HTML, що використовується в даному документі. Розміщується зверху документа.

<HEAD>...</HEAD> – заголовок HTML-документа.

<HTML>...</HTML> – контейнер, в який вміщується весь HTML-документ.

<META> – містить метаінформацію про властивості документа. Атрибути:

- http-equiv="..." – визначає тип властивості;
- value="..." – визначає властивість, значення: document –документ;
- content= "..." – задає значення властивості;
- url="..." – задає URL-адресу програми для властивості;
- name="..." – додатковий опис властивості. Якщо відсутній, то співпадає з http-equiv. Можливі значення: author –автор програми, author-corporate – фірма-розробник програми, keywords – ключ для пошуку (перелік слів), description – короткий опис Web-сторінки, призначений для відображення пошуковими системами.

Наприклад:

http-equiv ="content-Type" content="text/html; charset=windows-1251" – програма містить текст і коди HTML в кодуванні windows-1251.

http-equtv="content-language" content="en" – програма на англійській мові;

http-equiv="refresh" content="30" url="..." – сторінку з вказаною адресою завантажити через 30 секунд;

http-equiv="refresh" contents="30" – поточну сторінку перезавантажувати через кожні 30 секунд;

name="Keywords" content="список ключових слів" – список для пошуку по ключу даного документа пошуковими системами;

name="Description" content="короткий опис" – анотація – список для індексації документа пошуковими системами, приблизно, 200-300 символів. Звичайно в пошукових системах виводиться під назвою документа або спливає, якщо затримати покажчик миші на назві.

<SCRIPT>...</SCRIPT> – опис скрипта в заголовку. Атрибут language="..." – визначає тип скрипта.

<STYLE>...</ STYLE> – контейнер опису стилю. Має необов'язковий атрибут type="text/css", який визначає тип стилю.

<TITLE>...</TITLE> – контейнер призначений для визначення назви Web-

сторінки. Є обов'язковим елементом заголовка. Текст, укладений в контейнер, відображається в заголовку вікна браузера.

Теги форматування символів

<BASEFONT> – описує стандартний шрифт для даного Web-документа. Діє до наступного тега цього типу. Має наступні атрибути:

- face="..." – визначає гарнітуру шрифту або декілька можливих шрифтів, перелік значень обмежується встановленими на комп'ютері;
- size="..." – задає розмір шрифту;
- color="..." – задає колір шрифту.

<B>...</B> – виділити текст напівжирним шрифтом.

<BIG>...</BIG> – збільшує розмір шрифту щодо базового.

<CITE>...</CITE> – цитата, найчастіший текст, укладений в цей контейнер, виділяється курсивом.

<CODE>...</CODE> – машинописний текст, відображається шрифтом фіксованої ширини.

<DFN>...</DFN> – визначення, виділяється курсивом.

<EM>...</EM> – текст виділяється шрифтом із зрушенням і курсивом.

<Hn>...</Hn> – заголовок стилю n-го рівня (n = 1,2,..., 6). У стилі 1 – найкрупніший шрифт браузера, а в стилі 6 – найдрібніший шрифт. застосовується для структурування документа.

<FONT>...</FONT> – тег опису шрифту тексту, поміщеного в контейнер. Має наступні атрибути:

- face="..." – задає гарнітуру шрифту або декілька можливих шрифтів (Arial, Times New Roman,…);
- size= "..." – встановлює розмір шрифту. Визначені сім основних розмірів шрифту, що вимірюються не в пунктах, а в деяких умовних одиницях – від 1 до 7. Як правило, звичайний шрифт має розмір "3".
- color="..." – задає колір шрифту;
- style="..." – визначає опис стилю.

<PRE>...</PRE> – контейнер для тексту, відформатованого в іншому редакторі. Для його відображення застосовується тільки шрифт фіксованої ширини. Атрибут: width="..." – максимальне число символів в рядку.

<SMALL>...</SMALL> – текст, поміщений в даний контейнер, має розмір на одиницю менший, ніж основний текст.

<STRIKE>...</STRIKE> – текст, укладений в даний контейнер, перекреслюється горизонтальною лінією.

<STRONG>...</STRONG> – в даний контейнер поміщують важливий фрагмент документа. Зазвичай виділяється браузером напівжирним шрифтом.

_{...} – контейнер призначений для виведення нижніх індексів.

^{...} – контейнер призначений для виведення верхніх індексів.

<TT>...</TT> – текст, укладений в контейнер, відображається шрифтом фіксованої ширини.

<U>...</U> – підкреслений текст.

<VAR>...</VAR> – виділити змінні. Звичайно відображається шрифтом фіксованої ширини.

Теги форматування абзаців

<BLOCKQUOTE>...</BLOCKQUOTE> – контейнер для цитати. Зручно використовувати для створення полів в документі, тому що цитата зсувається вправо.

 – розрив рядку, текст, розташований після нього, переноситься на новий рядок без початку абзацу. Два оператори підряд дають пропуск рядка.

<CENTER>...</CENTER> – текст, укладений в цей контейнер, виводиться по центру документа.

<DIV>…</DIV> – контейнер абзацу. Атрибут align="..." задає вирівнювання тексту абзацу в рядку: center – вирівнювання по центру вікна браузера, left – по лівому краю, right – по правому краю, justify – по ширині.

<KBD>...</KBD> – текст вирівняний по ширині з обох боків.

<NOBR>...</NOBR> – вміщений текст не розбивається на рядки.

<P>...</P> – контейнер абзацу. Має ті ж атрибути, що і <DIV>

<SAMP>...</SAMP> – моделює текст, виведений на друк. Звичайно відображається шрифтом фіксованої ширини. Призначений для виведення прикладів програм, сценаріїв і т.д.

Теги створення списків

<DIR>...</DIR>, <UL>...</UL> або <MENU>...</MENU> – використовуються для формування маркірованого (ненумерованого) списку. Має атрибут type="...", який визначає тип маркера. Вид маркерів в кожному браузері свій для різних значень: disc (диск), square (квадрат), circle (круг). В якості маркерів, можуть бути використані графічні зображення. Кожен елемент списку визначається тегом <LI>.

<DL>...</DL> – список визначень. Теги елементів визначення:

<DT> – пояснюваний термін.

<DD> – визначення терміну. Обидва тега визначення можуть також використовуватися окремо для створення ненумерованих і не маркірованих списків.

<OL>...</OL> – контейнер нумерованого списку. Кожен елемент списку, як і для ненумерованого, визначається тегом <LI>. Атрибути:

- start="..." – задає початковий номер в списку;
- type="..." – встановлює тип маркера. Має наступні значення: A – маркер у вигляді прописних літер, a – малих літер, I – великих римських цифр, i – маленьких римських цифр, 1 – арабських цифр.

Теги розміщення гіперпосилань

<A>...</A> – контейнер для опису покажчика гіперпосилання. На покажчику потрібно клацнути мишею, щоб виконати перехід по посиланню. Може мати наступні атрибути:

- href="..." – URL-адреса документу (фрагменту, сторінки чи серверу), на який встановлюється посилання;

• name="..." – використовується для надання імені-ідентифікатора (аналог закладок редактора Word) фрагменту документа, на який встановлюється внутрішнє посилання;
• target="..." – встановлює параметри відкриття документа. Має наступні значення: _self завантажує сторінку в поточному вікні браузера; _blank – в новому вікні; _parent – у батьківський фрейм; _top – відміняє всі фрейми і завантажує сторінку в повному вікні браузера.

Теги вставки графічних елементів

<HR> – задає горизонтальну лінію. Можливі атрибути перераховані нижче:
• width="..." – довжина лінії в пікселях чи у відсотках від ширини вікна;
• size="..." – ширина лінії;
• color="..."– колір лінії;
• noshade – відміняє рельєфність лінії;
• align="..." – положення лінії на рядку екрану. Значення: left – притиснути лінію до лівого краю, center – розташувати по центру, right – притиснути до правого краю.

<IMG> – вставляє зображення на Web-сторінку. Атрибути:
• src="..." – URL-адреса файлу зображення;
• alt="..." – альтернативний текст, який виводиться замість малюнка, якщо малюнок не завантажився;
• height="..." і width="..." – встановлюють висоту і відповідно ширину зображення;
• border="..." – задає ширину рамки навколо зображення;
• vspace="..." і hspace="..."– визначають відстань по вертикалі і по горизонталі від зображення до краю тексту;
• align="..." – задає положення тексту щодо зображення. Значення left – текст обтікає малюнок, що розташовується ліворуч; right – відповідно праворуч; top, texttop –малюнок усередині текстового рядка, і його верхня границя збігається з верхньою границею літер; middle, absmiddle – малюнок розташовується усередині текстового рядка, що вертикально вирівнюється по центру малюнка; bottom, absbottom, baseline – малюнок розташовується усередині текстового рядка, і його нижня границя збігається з нижньою границею літер тексту.

<MARQUEE>...</MARQUEE> – контейнер рядка, що біжить. Текст, укладений в нього, переміщається по сторінці відповідно до наступних атрибутів:
• width="..." і height="..." – ширина і висота рядка, що біжить. Задається в пікселях, або у відсотках від ширини і висоти вікна;
• align="..." – задає положення зображення даного рядка по вертикалі;
• bgcolor="..." – встановлює фоновий колір рядка;
• behavior="..." – задає тип руху. Значення: scroll, slide, alternate;
• direction="..." – задає напрям руху. Значення: left (справа наліво),

right (зліва направо);

- scrollamount="..." і scrolldelay="..." – задають швидкість переміщення: число пікселів, на яке повинне переміститися зображення за задане число мілісекунд;
- loop="..." – задає число повторів анімації. За умовчанням встановлений режим безперервного повтору.

Теги створення табличних документів

<CAPTION> – контейнер заголовка таблиці. Має наступні атрибути: align="..." – задає місце заголовка, height="..."та width="..." – задає висоту та ширину блоку, в якому розташовується заголовок таблиці.

<TABLE>...</TABLE> – контейнер, в якому міститься таблиця. Має наступні атрибути:

- width="..." і height="..." – ширина і висота таблиці в пік селях, або у відсотках від ширини і висоти вікна;
- border="..." – ширина рамки. Якщо значення рівне нулю, то рамка не відображається;
- cellpadding="..." і cellspacing="..." – додають вільний простір між даними клітини і її межами, а також між елементами таблиці. Якщо рамка відсутня, то відступ від межі буде рівний сумі цих значень;
- align="..." – задає положення даних в таблиці по горизонталі.

<TD>...</TD> – контейнер елементу таблиці. Задані атрибути клітини пригнічують атрибути рядка і таблиці. Атрибути:

- width="..." і height="..." – ширина і висота в пік селях, або у відсотках від ширини і висоти вікна;
- align="..."– встановлює положення даних по горизонталі;
- valign="..." – встановлює положення даних по вертикалі, значення: bottom – вирівняти по нижньому краю, middle – розташувати по центру; top – притиснути до верхнього краю клітини;
- bgcolor= "..." – задає колір фону;
- bordercolor="..." – визначає колір рамки клітини;
- colspan="..." – задає число клітин, що об'єднуються в одну по рядку;
- rowspan="..." – встановлює число клітин, що об'єднуються в одну по стовпцю;
- nowrap – відміна перенесення слів на інший рядок усередині клітини.

<TH>...</TH> – контейнер визначення заголовка стовпця чи рядка. За умовчанням текст показується центрованим жирним шрифтом. Контейнер має такі ж атрибути, як і <TD> (див. вище).

<TR>...</TR> – контейнер рядка таблиці. Якщо контейнер порожній, то рядок залишається порожнім. Задані атрибути рядка мають більший пріоритет, ніж атрибути таблиці. Має такі ж атрибути, як і <TD>.

Теги розробки фреймів

<FRAME>...</FRAME> – контейнер опису окремого фрейма. Число таких тегів повинне бути рівне числу зарезервованих кадрів в контейнері (за

винятком вкладених контейнерів). Розташовуються кадри послідовно –зліва направо і зверху вниз. Тег має наступні атрибути:

- src="..." – URL-адреса вмісту кадру. Звично це файл HTML-документа з того ж каталогу, що і сам контейнер, але може бути і абсолютна адреса файлу з будь-якого комп'ютера. Якщо атрибут не заданий, кадр буде порожнім;
- marginwidth="..." і marginheight="..." – ширина від межі кадру до рамки по вертикалі і горизонталі;
- scrolling="..." – визначає смуги прокрутки. За умовчанням смуги прокрутки додаються в кадр, якщо вміст кадру більше його розмірів. Значення: yes – смуги прокрутки будуть обов'язково, no – смуги прокрутки заборонені, auto – смуги прокрутки додаються за умовчанням;
- noresize – забороняє користувачам змінювати розміри кадру. За умовчанням, якщо захопити рамку кадру мишею, з'являється двонаправлена стрілка, за допомогою якої можна перемістити рамку;
- frameborder="..." – задає тип рамки кадру. Значення: yes – рамка має об’ємну форму, no – рамка пласка;
- bordercolor="..." – задає колір рамки кадру;
- name="..." – ім'я кадру, показує, в який кадр слід виконувати завантаження при посиланні.

<FRAMESET>...</FRAMESET> – контейнер фреймів, тіло Web-сторінки у вигляді кадрів. Має наступні основні атрибути:

- rows= "...,..." – список значень висоти рядків зверху вниз;
- cols="...,..." – список значень ширини стовпців зліва направо;
- frameborder="..." – задає тип рамок між кадрами контейнера. Значення: no – рамки невидимі, тобто мають колір фону вікна браузера; yes – рамки кольорові;
- bordercolor="..." – задає колір рамок між кадрами. За умовчанням співпадає з кольором рамки вікна;
- border="..." – встановлює ширину рамок між кадрами контейнера. Якщо значення відсутнє, то кадри відділяються один від одного тонкими лініями заданого кольору;
- framespacing="..." – відстань між кадрами.

<NOFRAMES>...</NOFRAMES> – вкладений текст відображатиметься тільки в тому випадку, якщо браузер не підтримує механізм фреймів. Браузери, що підтримують кадри, ігнорують цей контейнер.

6.2.2. Спеціальні символи

Окрім тегів у мові HTML використовується спеціальний управляючий символ & – амперсант. Він застосовується для виведення спеціальних символів і символів з розширеної кодової таблиці, які не можна ввести з клавіатури.

При кодуванні використовується символьне (починається зі значка &) та цифрове позначення (починається з &#).

Наприклад, виведення самого символу амперсант (&) здійснюється за допомогою послідовності символів & або &, для виведення кутових дужок використовують послідовності < (для "<") і > (для ">"). У свою чергу, символ із заданим номером з кодової таблиці (наприклад 182) може бути заданий послідовністю ¶ .

Символами можуть крім значків, задаватися латинські та грецькі літери, та різні математичні позначення.

Найбільш поширені знаки, та їх коди приведені в таблиці 6.3.

Таблиця 6.3

Коди спеціальних символів HTML

Символ	**Код**	**Приклад**
Знак авторських прав	©	Copyright ©
Зареєстрована торгова марка	&red;	MMM®
Торгова марка	™	Company™
Менше	< або >	<
Більше	&gt,;	>
Ампресант	&	&
Нерозривний пропуск		
Тире	—	—
Лапка	"	„
Знак „собака”	@	@
Параграф	§	§
Плюс-мінус	±	±
Верхній індекс 1	¹	^{1}
Знак номера	№	№
Знак євро	€	€

Слід відмітити, що специфікація HTML 4.0 визначає код ™ для знаку торгової марки, проте, він ще не настільки поширений, як ™.

6.2.3. Позначення кольорів

Атрибути багатьох елементів HTML відповідають за визначення кольору елементів web-сторінки (фону, тексту, посилань).

Встановлення кольору можна проводити двома способами: завданням імені чи визначенням номера кольору по колірній схемі RGB (Red, Green, Blue).

Мова HTML підтримує наступні імена кольорів: AQUA, BLACK, BLUE, FUCHSIA, GRAY, GREEN, LIME, MAROON, NAVY, OLIVE, PURPLE, RED, SILVER, TEAL, WHITE, YELLOW.

Номер кольору RGB задається трьома двозначними шістнадцятирічними числами, причому кожне число належить інтервалу від 00 до FF і визначає інтенсивність відповідного кольору. Наприклад, номер кольору #FF0000 відповідає червоному кольору, оскільки має максимальну інтенсивність для червоного кольору, а зелений і блакитний мають значення, рівні нулю. Відповідно, номер

#00FF00 кодує зелений колір, а номер #0000FF – синій. Білий колір утворюється в результаті насичення всіх основних кольорів, тобто його кодом буде число #FFFFFF.

У разі повної відсутності всіх трьох кольорів можна одержати абсолютно чорний колір (код #000000). Очевидно, що, використовуючи таку комбінацію шістнадцятирічних чисел, можна одержати дуже велику палітру.

Фіолетовий, наприклад можна отримати за допомогою пари атрибут/значення color=”#EE82EE”.

Для зручності, основні шістнадцять кольорів, їх коди та назви приведені в таблиці 6.4.

Таблиця 6.4

Список шістнадцятирічних значень кольорів та їх текстових еквівалентів

Колір	**Код RGB**	**Ім'я**	**Колір**	**Код RGB**	**Ім'я**
чорний	#000000	black	срібний	#C0C0C0	silver
темно-бордовий	#800000	maroon	червоний	#FF0000	red
зелений	#008000	green	лимонний	#00FF00	lime
оливковий	#808000	olive	жовтий	#FFFF00	yellow
темно-синій	#000080	navy	синій	#0000FF	blue
фіолетовий	#800080	purple	фуксія	#FF00FF	fuchsia
чирок	#008080	teal	блакитний	#00FFFF	aqva
сірий	#808080	gray	білий	#FFFFFF	white

Кольорове оформлення сторінки надзвичайно важливий фактор в дизайні. Причому, над цим питанням працюють як професійні дизайнери, так і психологи.

В одному з випусків Internet журналу Internet Zone (http://www.izcity.com) приведений список комбінацій кольорів, розміщених в порядку погіршення сприйняття:

- синій на білому;
- чорний на жовтому;
- зелений на білому;
- чорний на білому;
- зелений на червоному;
- червоний на жовтому;
- червоний на білому;
- оранжевий на чорному;
- оранжевий на білому;
- червоний на зеленому.

Колір тексту обов’язково повинен бути узгоджений з кольором фону. Заголовки та гіперпосилання, окремі текстові блоки, можна виділяти окремо, використавши колір символів, чи встановивши для них певний фон.

Схема кольорів повинна приваблювати око, в той же час не повинна бути втомливою і нав'язливою.

6.2.4. Одиниці виміру в HTML

В HTML використовуються декілька типів одиниць виміру:

1) Абсолютні:
   а) Реальні одиниці використовуються досить рідко, коли точно відомі параметри браузера для перегляду сторінки:
      - in – розмір в дюймах, 1 дюйм дорівнює 2,54 сантиметри;
      - cm – розмір в сантиметрах;
      - mm – розмір в міліметрах;
   б) Поліграфічні одиниці, що застосовуються в поліграфії для встановлення кегля шрифту:
      - pt – пункт, саме в них визначається розмір шрифту в текстових редакторах, приблизно дорівнює 0,36см;
      - pc – піка, дорівнює 12 pt.
2) Відносні, встановлюються відповідно до деякого відомого значення:
   а) em – розмір відносно ширини стандартної типографської літери „m", значення 3m, встановить розмір шрифту втричі більше базового;
   б) ex – розмір базується на висоті літери „x" базового шрифту;
   в) px – розмір в пікселях пристрою відображення, тобто зображення залежить від роздільної здатності монітора і параметрів браузера;
   г) % – розраховується відносно розмірів елементів та інших об'єктів: ширина лінії відносно ширини вікна браузера; встановлення розміру малюнка відносно його реальних розмірів;

Однозначних рекомендацій по використанню тих чи інших одиниць виміру немає. Тут скоріше потрібно керуватися власним досвідом і логікою. Наприклад, для визначення ширини лінії, краще використовувати відсотки, а товщини пікселі.

За замовченням всі розміри в HTML-коді встановлюються в пікселях. Тобто, атрибут width="100" трактується як ширина в 100 пікселів.

6.2.5. Приклади Web-сторінок

Наведемо декілька прикладів, які пояснюють роботу різних тегів по представленню тексту файлів у HTML-кодах та їх вигляд у браузері.

Форматування тексту.

Форматування тексту включає: форматування символів, тобто їх розмір, стиль, колір, накреслення; розмітку абзаців, а саме їх визначення, вирівнювання та відступи; задання різноманітних різнорівневих списків.

Параметри шрифту, який використовується для відображення тексту на web-сторінках, найпростіше визначити за допомогою тегу <FONT>, атрибути якого визначають спосіб відображення тексту. Результати застосування різних атрибутів показано на прикладі 1 (рис. 6.2).

Необхідно розрізняти записи <FONT SIZE="+1" і <FONT SIZE="1">. В першому випадку вказується відносний розмір шрифту, а в другому — абсолютний. Можна використовувати також значення "+2", "-2", "+3" і т. д.

```
<html>
<head>
<title> Оформлення символів </title>
</head>
<body>
<font color="#0000ff" face="Arial Black" size="+3"> Особливості написання тегів.</font><br>
<div align="justify">
<font face="Times New Roman" size="5pt"> Комбінація з відкриваючого і закриваючого тега називається <font face="Century Gothic"> контейнером тегів. </font> "Вмістом тега" може бути текст чи інші теги.</div>
<div align="justify">
Теги не чутливі до регістра. </div>
<div align="justify"> Теги не чутливі до їх розташування на сторінці. </font>
</div>
</body></html>
```

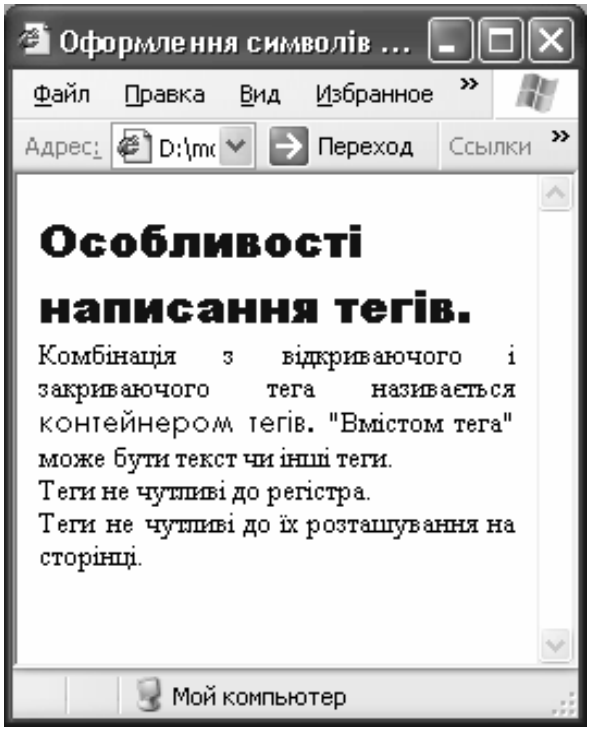


Рис. 6.2. Приклад форматування символів

Другий приклад (рис. 6.3) ілюструє заголовки різних розмірів від H1 до H6, які можуть використовуватися для структурування тексту документу через багаторівневі заголовки.

Далі показано формування маркірованих (тег UL) та нумерованих (тег OL) списків. Кожен елемент списку починається тегом LI. Він по своїй структурі парний, але закриваючий тег використовувати не обов’язково.

Тег P переводить текст на наступний рядок, B – робить шрифт жирним, I – курсивом, U – підкреслює текст, який в ньому знаходиться. Всі ці теги можна розташувати один в однім, але потрібно уважно ставитись до порядку їх закриття.

Для виділення абзаців можна використовувати дві основних теги – P чи DIV. Перший, використовує американський метод форматування, відділяючи абзаци пустим рядком. DIV – більше орієнтований на звичне українське форматування.

Тег PRE дозволяє вставляти текст, який було відформатовано раніше якимось текстовим редактором. Це буває корисним, наприклад, при цитуванні віршів або статей, розташування тексту в яких має принципове значення.

```
<HTML>
<HEAD>
<TITLE> Приклад заголовків, списків, параграфів </TITLE>
</HEAD>
<BODY>
<H1> Заголовок1 </H1>
<H2> Заголовок2 </H2>
<H3> Заголовок3 </H3>
<H4> Заголовок4 </H4>
<H5> Заголовок5 </H5>
<H6> Заголовок6 </H6>
 <UL>
<LI> Рядок 1 списку
<LI> Рядок 2 списку
<LI> Рядок 3 списку
</UL>
<OL>
<LI> Рядок 1 списку
<LI> Рядок 2 списку
<LI> Рядок 3 списку
</OL>
<P> Текст <B>Жирний</B>
 <I>Курсив</I>
 <U>Підкреслений</U></P>
<PRE>
Тек  ст  нефор
  матова  ний
</PRE>
</BODY>
</HTML>
```

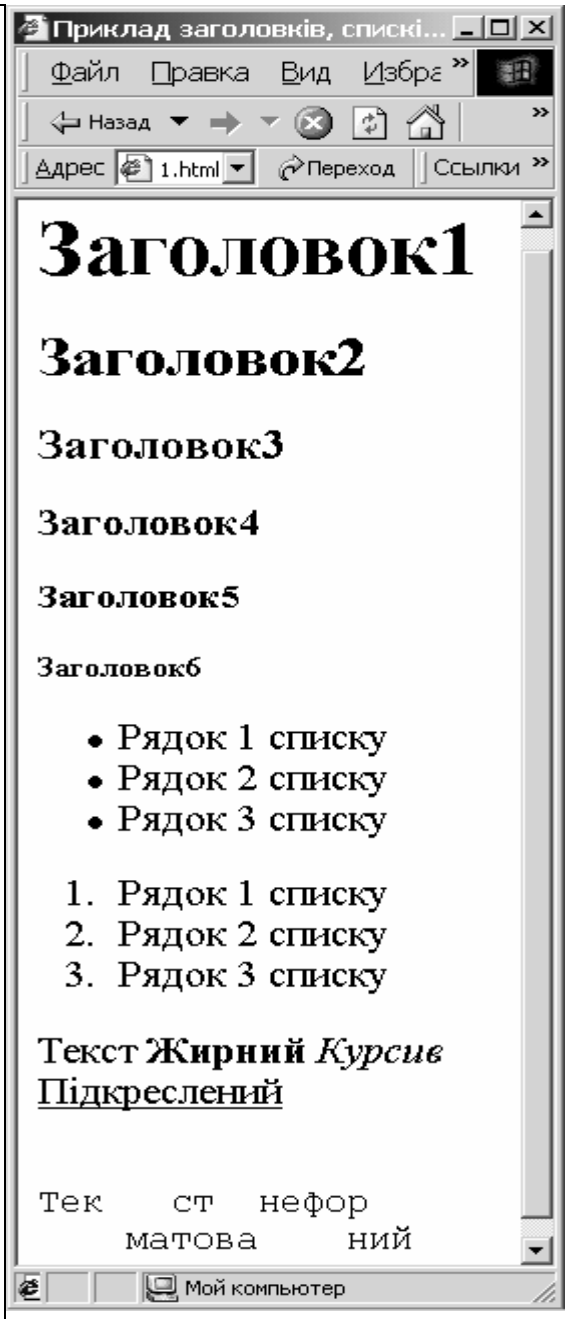


Рис. 6.3. Приклад, заголовків, списків та параграфів

При бажанні можна розроблювати різноманітні ієрархічні списки, комбінуючи маркіровані та нумеровані списки різних типів (рис. 6.4). Головне при цьому, чітко уявляти вкладеність основних елементів.

Розробка таблиць.

Одним з основних елементів web-сторінок є таблиці. Таблиці у Web-документах застосовуються не тільки для впорядкування числових даних, а й для вставки зображень і посилань, для раціонального компонування Web-сторінок.

Таблиці допомагають відійти від ієрархічного розміщення тексту на Web-сторінках, за необхідності можна зробити їх границі невидимими.

```
<html>
<head>
<title>Ієрархічні списки</title>
</head>
<body>
<ol>
<li>За типом засобів комунікації:
<ul type="disc">
  <li>наземні багатовузлові мережі;
  <li>супутникові радіомережі;
  <li>комбіновані мережі.</ul>
<li>За способом комутації повідомлень:
<ul type="disc">
  <li>комутація каналів;
  <li>комутація повідомлень;
  <li>комутація пакетів;
  <li>адаптивна комутація.</ul>
<li>За вибором маршруту передачі повідомлення:
<ul type="disc">
  <li>фіксовані шляхи;
  <li>спрямований вибір шляху;
  <li>випадкові шляхи;
  <li>лавинний спосіб.
</ul>
</ol>
</body></html>
```

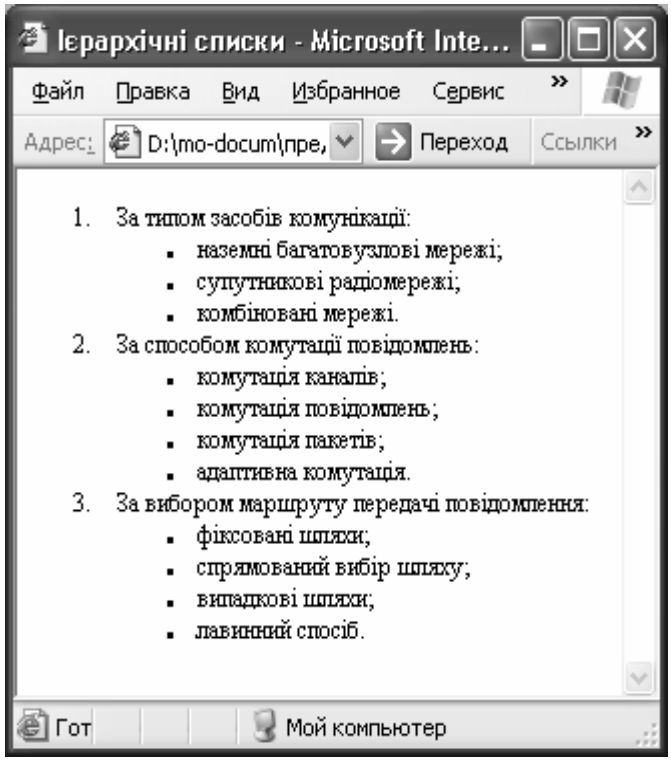


Рис. 6.4. Приклад оформлення ієрархічних списків.

Таблиці будуються за принципом вкладення і вводяться на сторінці за допомогою ряду елементів. Кожна таблиця починається тегом <TABLE> і закінчується </TABLE>. Створювана таблиця ніби розгортається по рядках, а рядки заповнюються чарунками.

Наведемо приклад складної таблиці з вирівнюванням елементів різними стилями.

Тег TH дає заголовок таблиці, теги TR і TD формують рядки і колонки таблиці.

Інколи в таблицях зустрічаються так звані *об'єднані чарунки* — коли декілька розміщених поряд чарунок зливаються в одну велику. Режим тега TH – COLSPAN – забезпечує використання одразу двох колонок таблиці під заголовок. Режим тега TD – ROWSPAN – забезпечує використання одразу двох рядків під клітинку таблиці.

Цей код дає наступне зображення (рис. 6.5).

```
<HTML>
<HEAD>
<TITLE> Приклад таблиці </TITLE>
</HEAD>
<BODY>
<TABLE BORDER =”1” CELLPADDING= “3” BGCOLOR= “white” >
<TH> Мова програмування </TH>
<TH ALIGN =”right” COLSPAN =”2”> Особливості </TH>
<TR>
<TD> C++ </TD>
<TD ROWSPAN = “2”> Використовує </TD>
<TD> Класи </TD>
<TR>
<TD> Visual BASIC </TD>
<TD> Об`єкти </TD>
</TABLE>
</BODY>
 </HTML>
```

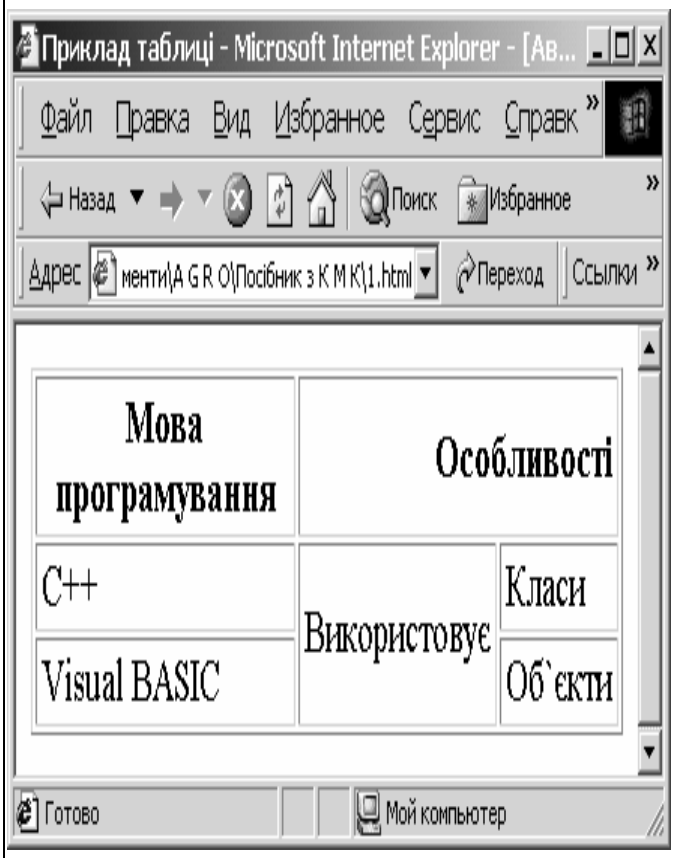


Рис. 6.5. Приклад зображення таблиці

Розглянуті таблиці являються статичними, тобто в таблицях HTML не передбачене автоматичне виконання математичних функцій.

Для того, щоб пересвідчитись, що в більшості сучасних Web-сторінок використовується табличне представлення елементів, досить відкрити знайомий вузол (наприклад, www.mail.ru) і переглянути HTML-код.

Вставка гіперпосилань та рисунків.

Гіперпосилання – контекстові зв’язки між розміщеними в Internet матеріалами. Вони є основою структури World Wide Web. Користувачам зазвичай подобаються сторінки насичені гіперпосиланнями, за допомогою яких зручно отримати більш докладну інформацію. Будь-яке слово, розміщене на Web-сторінці, можна перетворити в гіперпосилання, якщо відомі інші сторінки в мережі, більш широко розкриваючи даний предмет. Клацаючи кнопкою миші по гіперпосиланням, можна в обійти всю WWW.

В текст Web-сторінки можна вставляти гіперпосилання трьох типів:

- посилання в інші місця поточної сторінки, в якості значення атрибута HREF= вказують ім’я певного наперед визначеного за допомогою атрибута NAME місця сторінки;
- посилання на інші сторінки цього же сайту чи сервера, в якості значення атрибута HREF= вказують ім’я файла;
- посилання на сторінки, розміщені на будь-якому серверу в Internet, в якості значення атрибута HREF= вказують повну URL-

адресу ресурсу.

Розглянемо приклад застосування тега <A> для різних варіантів переходів (рис. 6.6). Режим NAME задає ім'я помітки на яку можна перейти, якщо в іншому тегу <A> встановити режим href="# ім'я помітки".

```
<HTML>
<HEAD>
<TITLE> Вставка посилань і малюнків
</TITLE>
</HEAD>
<BODY>
Перехід в інше місце документу
<A NAME="HTML0"></A>
<P>Перехід до іншого документу</P>
<P>
<A href="h2.html">Перехід до сторінки
h2.html </A>
 </P>
<P>Вивести малюнок на сторінку </P>
<P> <IMG SRC="email.jpg" ALT="Це
малюнок поштової скриньки" >
</P>
<A href="HTTP://WWW.HOME.COM
">
<IMG SRC="email.jpg" width=10
length=10>
<A href="mailto: kadet2@mail.ru">
Перейти на мою поштову адресу
</A>
<P>
<A href="#HTML0">
Перейти на помітку. </A>
</P>
</BODY>
</HTML>
```

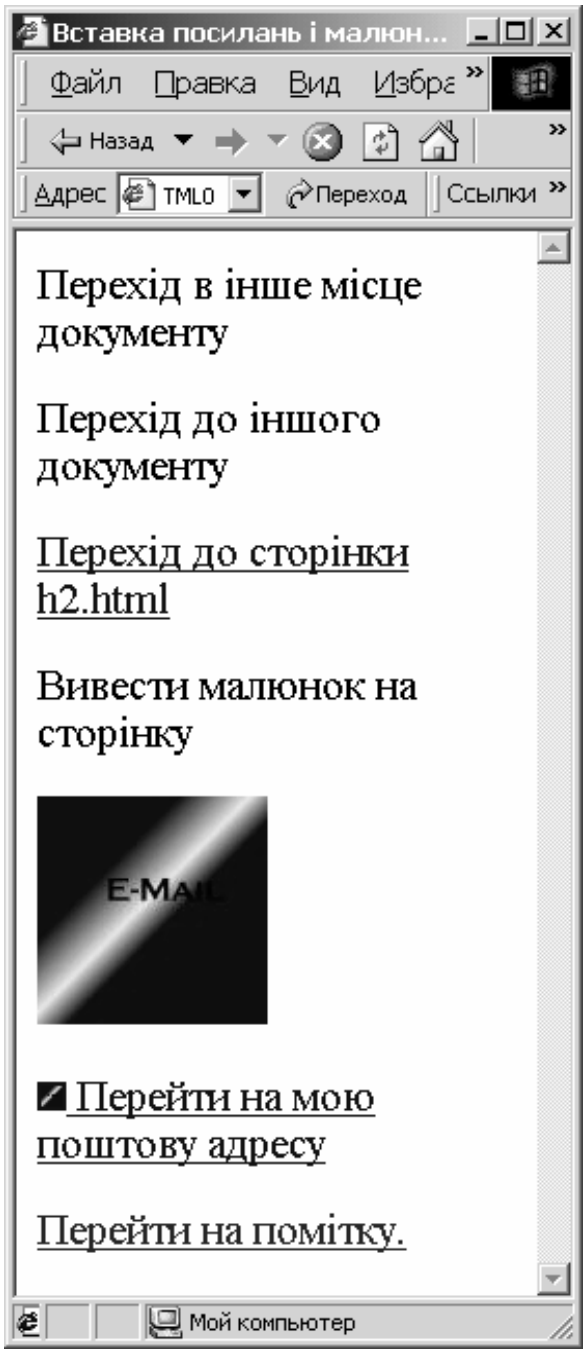


Рис 6.6. Приклад вставки гіперпосилань і малюнків

Перехід до іншого документу (іншої web-сторінки) здійснюється теж через цей режим href=" " – там треба вказати шлях до іншого html-файла.

Якщо в тегу <A> вказати режим href="mailto: " і далі написати свою електронну адресу, комп'ютер автоматично викличе програму-мейлер, яка встановлена на ньому, і вкаже вашу адресу в полі адресації ТО.

Зображення на Web-сторінці можуть надавати певну інформацію або ство-

рювати загальний настрій, „оживляючи” простий текст. Найбільш розповсюджені випадки застосування зображень:

- логотип компанії;
- графіка для рекламного повідомлення;
- різноманітні малюнки;
- діаграми та графіки;
- художні шрифти;
- підпис автора сторінки;
- застосування графічних елементів для створення роздільної лінії;
- використання графічних маркерів для створення гармонійних маркірованих списків.

Існує безліч різних форматів для представлення графіки. Але, у якості стандартних для використання в Internet прийняті тільки три з них. Це *GIF* (скорочення від Graphics Interchange Format – графічний формат обміну), JPEG(названий по імені групи творців — *Joint Picture Expert Group)* і порівняно новий формат *PNG (Portable Network Graphics).*

На швидку стандартизацію претендують формати *DjVu* (цей формат оптимізований для представлення документів, що містять одночасно текст і графіку) і *LuRaTech Wavelet (LWF)* — формат, що відрізняється можливістю високого ступеня стиску при досить високій якості і можливістю заздалегідь установити розмір майбутнього файлу.

Однак на даний час краще орієнтуватися лише на формати — GIF і JPEG. Вони підтримуються всіма броузерами і не вимагають яких-небудь додаткових модулів для відображення. Ці формати були створені для збереження графіки в стиснутому виді.

Для того, щоб вивести малюнок на сторінку, потрібно скористатися тегом IMG, в якому режим SRC= задає адресу місця знаходження цього малюнка.

Можна також здійснювати будь-які переходи за допомогою тега <A> не з тексту, а з малюнка. Для цього замість тексту потрібно вставити тег IMG з відповідним режимом SRC=.

Гарним тоном є наявність так званого альтернативного тексту для тих випадків, коли в броузері відключений перегляд графіки. Тоді на місці рисунку виводиться прямокутник відповідного розміру, з поясненням, що за картинка повинна бути на цьому місці. Альтернативний текст уводиться як значення атрибута ALT.

Створити або підготувати існуючий малюнок для вставки на сторінку можна за допомогою спеціальних графічних редакторів, наприклад Adobe Photoshop, Corel Draw, Xara Webstyle і т.п.

При розробці сайту зазвичай створюється спеціальна папка, наприклад \Images, в яку поміщаються файли всіх графічних елементів, що розміщуються на сторінках.

Броузер Internet Explorer (версія 4 і вище) дозволяє за допомогою тегу <IMG> завантажувати і переглядати також відео фрагменти у форматі AVI (Video for Windows). Для цього можна установити атрибут DYNSRC=”ім'я AVI-

файлу”. При цьому можна додатково вказати в атрибуті START=, коли варто починати програвання відео: "FILEOPEN" — відразу після завантаження файлу або "MOUSEOVER" — після наведення покажчика миші на зображення. Атрибут LOOP= дозволяє установити кількість програвань: "0" означає нескінченне повторення, а будь-яке інше число вказує точну кількість повторів.

Використання фреймів.

Фрейми (frames – кадри) розбивають Web-сторінку на незалежні вікна, що містять власну інформацію. В кожне з цих вікон завантажуються окремі HTML-документи.

За допомогою фреймів сторінка може бути розділена, наприклад на дві області, в одній буде розміщуватися навігація по сайту, а в іншій – завантажуватися документи, вибрані користувачем (рис. 6.7).

index.html

```
<HTML>
<HEAD>
<TITLE> Приклади застосування фреймів </TITLE>
</HEAD>
<FRAMESET cols="140,*">
<FRAME src="menu.html" name="MENU">
<FRAME src="content.html" name="content">
</FRAMESET>
</HTML>
```

menu.html (основна частина)

```
<A href="font.html" target="content"> Приклад заголовків, списків, параграфів </A><BR><BR>
<A href="hyper.html" target="content"> Вставка посилань і малюнків </A><BR><BR>
<A href="table.html" target="content"> Приклад таблиці </A
```

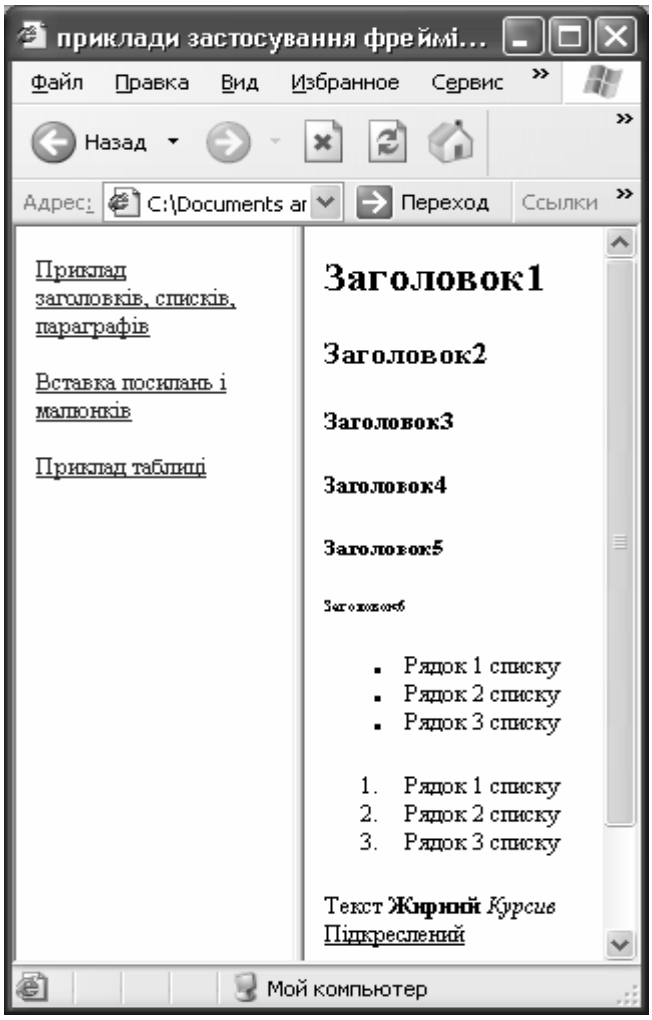


Рис 6.7. Приклад застосування фреймів

Серед достоїнств фреймів слід зазначити: простоту використання; швидкість верстки і завантаження; розміщення в чітко визначеному місці браузера; зміна розмірів областей.

Крім достоїнств фрейми мають і ряд недоліків: навігація – при переході по посиланні, заголовок сторінки не змінюється; погана індексація пошуковими системами; несумісність з деякими браузерами; непрестижність, професіонали

рідко використовують фрейми.

Для створення фрейму, як уже зазначалося, використовується елемент <FRAMESET>, який заміняє елемент <BODY> в документі і використовується для розділу екрану на області-фрейми.

Всередині елемента знаходяться елементи <FRAME>, які посилаються на HTML-документ, призначений для завантаження в область фрейма.

Об'єднаємо попередні приклади в загальну структуру. Стартова сторінка буде називатися index.html. Зліва на сторінці, розміщується меню, тобто в нашому випадку це назви трьох прикладів, справа – виводитись вміст вибраного документа.

6.3. Таблиці стилів CSS

Вперше ідея форматування документів за допомогою каскадної таблиці стилів (Cascading Style Sheets – CSS) була рекомендована Консорціумом W3C в 1996 році. З появою CSS у розробників сайтів з'явилась можливість відокремити структуру HTML- документа від його формату.

CSS (Cascading Style Sheets – каскадні листи стилів) є інструментом впливу на зовнішній вигляд Web-сторінок.

Стилі – це набір властивостей, які можна привласнити багатьом елементам (таким як таблиці, текст посилання і т.п), і які впливають на відображення об'єкта у вікні браузера.

Мова CSS є мовою високого рівня і використовує стандартну термінологію, прийняту в електронній поліграфії. За допомогою CSS можна встановити відстань між символами, міжрядкові інтервали, колір тексту і фону, розмір і гарнітуру шрифту і т.п.

Таблиці стилів спрощують розмітку HTML, задаючи для основних структурних елементів сторінки свої стилі. Наприклад: заголовок – червоний колір, розмір в два рази більше звичайного, вирівнювання по центру; посилання – шрифт Arial, колір зелений і т.п.

CSS передбачає три типи таблиць стилів:

- вбудована – властивості стиля задаються в верхній частині документа і застосовуються для всіх елементів HTML- документу;
- внутрішня – атрибути стиля можуть бути визначені в будь-якій частині Web-документа для окремого фрагмента коду;
- зовнішня – властивості стилів знаходяться в окремому файлі.

Таблиці CSS мають наступні особливості:

- Гнучкість підключення. Інформація про стиль може бути розміщена практично в будь-якому місці документа чи в окремому файлі.
- Каскадність. Передбачає, що стилі можуть описані в різних місцях, що утворює своєрідний „каскад" правил, якими повинен керуватись браузер при відображені відповідних елементів. Каскадність передбачає також пріоритет стилів, підключених пізніше, над стилями, підключеними раніше.

- Залежність від середовища представлення. Таблиці застосовуються до конкретних носіїв, тобто вони можуть визначити за допомогою скриптів роздільну здатність монітора комп'ютера користувача та тип використовуваного браузера і підключити відповідну таблицю стилів для найкращого відображення змісту.
- Спадковість. Перенесення форматування елементів на вкладених в них. Причому, при зміні стилю батьківського елемента, автоматично змінюються стилі дочірніх.

6.3.1. Базові поняття. Специфікація CSS

Розробник повинен мати базові знання по HTML і по поліграфічній термінології. Синтаксис правила стиля CSS в найбільш загальному вигляді має наступну форму:

селектор {визначення}

Визначенням стилю називається пара значень „властивість: значення".

Наприклад, для визначення кольору елементів 'H1' як синій, достатньо написати:

```
H1 { color: blue }
```

Наведений приклад є простим правилом CSS. Хоча він намагається вплинути тільки на одну з властивостей, необхідних для побудови HTML документа, він вже є таблицею стилів. Скомбінований з іншими таблицями стилів (однією з фундаментальних властивостей CSS є комбінування таблиць стилів) він визначатиме остаточний вигляд всього документа.

Селектор є зв'язком між HTML документом і таблицею стилів, повний набір типів елементів містить всі можливі селектори. Типи елементів визначені в специфікації HTML.

Авторам Web-сторінок необхідно визначати свою таблицю стилів тільки в тому випадку, якщо вони хочуть запропонувати специфічний стиль для своїх документів.

Текстові коментарі в таблицях стилів оформляються так:

```
 EM { color: red }  /* червоне, справді червоне!! */
```

Коментарі не можуть вкладатися один в одного. Для обробника CSS коментар еквівалентний пропуску.

6.3.2. Включення в HTML

Для того, щоб таблиця стилів впливала на вигляд документа, браузер повинен знати про її існування. Специфікація HTML визначає способи включення таблиць стилів в HTML. Тому даний розділ є інформативним, а не нормативним:

```
<HTML>
 <HEAD>
  <TITLE>title</TITLE>
        <LINK REL=STYLESHEET TYPE="text/css"
```

```
                    HREF="http://style.com/cool" TITLE="Cool">
                      <STYLE TYPE="text/css">
                                    @import url(http://style.com/basic);
                            H1 { color: blue }
                      </STYLE>
  </HEAD>
  <BODY>
          <H1>Headline is blue</H1>
          <P STYLE="color: green">While the paragraph is green.
  </BODY>
</HTML>
```

Даний приклад демонструє чотири способи об'єднання стилю з HTML:

- використовуючи елемент LINK для зв'язку із зовнішньою таблицею стилів, посилається на альтернативну таблицю стилів, яку може вибрати читач, тоді як таблиці стилів, що імпортуються, автоматично об'єднуються з рештою таблиць стилів;
- використовуючи елемент STYLE усередині елементу HEAD;
- імпортуючи таблицю стилів за допомогою нотації CSS @import;
- використовуючи атрибут STYLE в елементі усередині секції BODY. Цей спосіб змішує стиль з вмістом і тому втрачає відповідні достоїнства традиційних таблиць стилів.

Традиційно браузери ігнорували невідомі теги. Тому старі ігноруватимуть елемент STYLE, але його вміст вважатиметься, частиною тіла документа і відображатиметься в документі. Протягом перехідного періоду, вміст елементу STYLE можна "ховати", використовуючи коментарі SGML:

```
<STYLE TYPE="text/css">
<!--
  H1 { color: green }
-->
</STYLE>
```

6.3.3. Групування

Для зменшення розміру таблиці стилів, селектори можна групувати в списки, розділені комами:

```
H1, H2, H3 { font-family: helvetica }
```

Точно також можна групувати визначення:

```
  H1 {
          font-weight: bold;
          font-size: 12pt;
          line-height: 14pt;
          font-family: helvetica;
          font-variant: normal;
          font-style: normal;
```

```
}
```

Зазначимо, що деякі властивості мають власний синтаксис групування:

```
H1 { font: bold 12pt/14pt helvetica }
```

що еквівалентне попередньому прикладу.

6.3.4. Успадкування

У першому прикладі був встановлений синій колір елементів в H1. Уявіть, що є елемент H1 має всередині елементом <EM>:

```
<H1>The headline <EM>is</EM> important!</H1>
```

Якщо для елементу EM не було визначено ніякого кольору, то параметр "is" успадкує колір батьківського елементу, тобто буде відображений синім кольором. Інші властивості стилю також спадкуються, наприклад 'font-family' і 'font-size'.

Для визначення властивості стилю для документа за умовчанням, можна задати цю властивість елементу, від якого успадковується решта видимих елементів. В HTML-документах цю функцію виконує елемент BODY:

```
BODY {
        color: black;
        background: url(texture.gif) white;
                                                }
```

Цей приклад спрацює, навіть якщо автор опустив тег BODY (що є допустимим) тому, що браузер самостійно вставить пропущений тег. Попередній приклад встановлює чорний колір тексту, а фон – містить картинку. Якщо рисунок буде недоступний, фон залишиться білим.

Деякі властивості стилів не успадковуються від батьківських дочірніми елементами. В більшості випадків інтуїтивно зрозуміло коли виникає така ситуація. Наприклад властивість 'background' не успадковується, але фон батьківського елементу за умовчанням просвічуватиме скрізь.

Інколи значення властивості указується у відсотках від іншої властивості:

```
P { font-size: 10pt }
P { line-height: 120% }  /* relative to 'font-size', i.e. 12pt */
```

Для будь-якої властивості, значення якої можуть указуватися у відсотках, визначено властивість на яку вона посилається. Дочірні елементи P успадкують обчислене, а не процентне значення 'line-height' (12pt).

6.3.5. Клас як селектор

Для збільшення гнучкості контролю над елементами, в HTML доданий новий атрибут CLASS. Всі теги усередині елементу BODY можуть бути класифіковані, а на клас можна послатися через таблицю стилів:

```
<HTML>
  <HEAD>
      <TITLE>Title</TITLE>
      <STYLE TYPE="text/css">
```

```
                H1.pastoral { color: #00FF00 }
            </STYLE>
        </HEAD>
        <BODY>
            <H1 CLASS=pastoral>Way too green</H1>
        </BODY>
    </HTML>
```

До класифікованих елементів застосовуються звичайні правила спадкоємства, вони успадковують значення своїх батьків в структурі документа.

Можна адресувати всі елементи одного класу опустивши ім'я тега в селекторі:

```
.pastoral { color: green }  /* all елементи with CLASS pastoral */
```

Для кожного селектора можна визначити тільки один клас. Тому P.pastoral.marine неправильний селектор в CSS. Контекстові селектори можуть визначати один клас для одного простого селектора.

CSS надає настільки великі можливості для використання атрибуту CLASS, що у багатьох випадках не важливо для якого елементу HTML встановлюється клас, можна примусити будь-який елемент емулювати будь-який інший елемент. Але не рекомендується покладатися на таку можливість, оскільки це позбавляє документ структури, яка має універсальне значення. Структура заснована на класах, має вузьке застосування тільки в тих випадках, коли значення класів обумовлені з обох боків.

6.3.6. ID як селектор

HTML визначає атрибут ID, який має унікальне значення в документі. Він важливий, як селектор таблиці стилів, і може адресуватися за допомогою '#':

```
#z98y { letter-spacing: 0.3em }
H1#z98y { letter-spacing: 0.5em }
<P ID=z98y>Wide text</P>
```

У приведеному прикладі перший селектор відповідає елементу P завдяки ID-атрибуту. Другий селектор визначає як тип елементу (H1), так і значення ID, і, тому, не відповідає елементу P.

Використовуючи ID як селектор, можна встановлювати властивості поелементно. Тоді як таблиці стилів були розроблені для візуалізації структури документа, ця властивість дозволяє авторам створювати документи, які коректно відображаються без використання структурних елементів HTML. Але таке застосування таблиць стилів не вітається.

6.3.7. Контекстові селектори

Спадкоємство значно зменшує об’єми тексту CSS. Замість того, щоб указувати всі властивості стилів, можна встановити значення за умовчанням, а потім вказати виключення. Для того, щоб елементи EM усередині H1 мали інший текст, можна було б вказати:

```
H1 { color: blue }
EM { color: red }
```

При використанні цієї таблиці стилів, весь текст усередині EM і поза H1, стане червоним. Очевидно, хотілося, щоб тільки ті елементи EM, які укладені в H1 стали червоними. Цього можна досягти, вказавши:

```
H1 EM { color: red }
```

В цьому випадку селектор є маскою пошуку в стеку відкритих елементів, а такий селектор називається контекстним селектором. Контекстні селектори складаються з простих селекторів, розділених пропуском (всі описувані до цього селектори були простими селекторами).

Описані правила застосовуються тільки до елементу, який відповідає останньому простому селектору (в даному випадку елемент EM), і лише в тому випадку, якщо результат пошуку є позитивним. Контекстові селектори в CSS описують тільки спадкові взаємозв'язки, тоді як подальші версії можуть описувати інші види зв'язку.

```
UL LI    { font-size: small }
UL UL LI { font-size: x-small }
```

В цьому випадку перший селектор відповідає елементам LI з як мінімум одним предком UL. Другий селектор відповідає підмножині першого, тобто елементи LI з як мінімум двома предками UL. Конфлікт вирішується тим, що другий селектор є більш специфічним у зв'язку з довшою маскою пошуку.

Контекстові селектори можуть містити тип елементу, атрибути CLASS, атрибути ID або їх комбінацію:

```
DIV P          { font: small sans-serif }
.reddish H1    { color: red }
#x78y CODE     { background: blue }
DIV.sidenote H1 { font-size: large }
```

Перший селектор відповідає елементам P, які серед предків мають DIV. Другий селектор відповідає всім елементам H1 які мають предка класу reddish. Третій селектор відповідає всім елементам CODE, які є спадкоємцями елементу з ID=x78y. Четвертий селектор відповідає всім елементам H1, які мають предка DIV з класом sidenote.

Можна групувати декілька контекстових селекторів:

```
H1 B, H2 B, H1 EM, H2 EM { color: red }
```

що еквівалентне:

```
H1 B { color: red }
H2 B { color: red }
H1 EM { color: red }
H2 EM { color: red }
```

6.3.8. Популярні властивості CSS

Як уже відмічалося, стилі визначають параметри відображення на сторінці. В таблиці 6.5 приведено найчастіше застосовувані властивості та їх можливі значення.

Таблиця 6.5

Властивості каскадних таблиць стилів

Власти-вість	Значення	Опис	Приклад
Властивості шрифту CSS			
font-family		*Гарнітура шрифту*	p{font-family:Arial, sans-serif}
	Arial	Визначає список гарнітур шрифту в порядку зменшення пріоритету	
font-style		*Накреслення шрифту*	p{font- style: italic}
	normal	Звичайний текст	
	italic	Курсивний шрифт	
	oblique	Косий шрифт	
font-variant		*Регістр шрифту*	p{font-variant: small-caps}
	normal	Нормальний	
	small-caps	Малі прописні літери	
font-weight		*Товщина шрифту*	p{font-weight: bold}
	normal	Нормальна жирність	
	lighter	Світліше	
	bold	Напівжирний	
	bolder	Жирніше	
	100-900	100 – світлий, 900 – самий жирний	
font-size		*Розмір шрифту*	p{font-size:12pt}
	normal	Нормальний розмір	
	pt	Пункти	
	px	Піксели	
	%	відсотки	
Властивості тексту CSS			
line-height		*Міжрядковий інтервал*	
	normal	Нормальний	line-height: nor-mal
	множник	Висота рядку – добуток множника на розмір шрифту	line-height:1.5
	Одиниці довжини	Задається точно	line-height:12px
	%	Відсотки від розміру шрифту	line-height:120%
text-decoration		*Оформлення тексту*	text-decoration: none
	none	Оформлення відсутнє	
	underline	Підкреслення	
	overline	Лінія над текстом	
	line-through	Закреслення тексту	
	blink	Мигтіння тексту	

Власти-вість	Значення	Опис	Приклад
text-transform		*Регістр тексту*	text-transform: lowercase
	none	Регістр не переключається	
	capitalize	Перші літери змінюються на прописні	
	uppercase	Всі прописні	
	lowercase	Всі малі	
text-align	*Вирівнювання тексту в блочних елементах*		text-align: justify
	left	По лівому краю	
	right	По правому краю	
	center	По центру	
	justify	По ширині	
letter-spacing	Одиниці довжини	Відстань між літерами (можна використовувати від'ємні значення)	letter-spacing:5px
white-space		*Робота з пропусками*	white-space: pre
	normal	Звичайний режим – зайві пропуски ігноруються, рядки переносяться автоматично	
	pre	Всі пропуски враховуються	
	nowrap	Заборонений перенос рядків, зайві пропуски ігноруються	
text-indent		*Відступ першого рядку*	text-indent:15px; text-indent: 10%
	Одиниці довжини	Відступ в одиницях довжини	
	%	Відступ у % до ширини блоку	
Властивості CSS для управління кольором і фоном			
color	колір	*Задає колір тексту*	p{color:#999999 }
back-ground-color		*Колір фону*	BODY{ back-ground-color:#888888}
	колір	Суцільний колір	
	transparent	Прозорий фон	
back-ground-image		*Фоновий малюнок*	BODY{ back-ground-image: url(pict/bg.gif}
	URL	URL малюнка	
	none	Немає малюнка	
back-ground-repeat		*Повторюваність фонового малюнка*	BODY{ back-ground-repeat: repeat-y}
	repeat	Повторює малюнок по гор. і верт.	
	repeat-x	Повтор тільки по горизонталі	
	repeat-y	Повтор тільки по вертикалі	
	no-repeat	Не повторює малюнок	
back-ground-	*Прокрутка фона разом з документом*		BODY{backgrou nd-image:url pic-
	scroll	Фон прокручується	

Властивість	Значення	Опис	Приклад
attachment	fixed	Фон не прокручується	tures/ bg.gif background-attachment: fixed}
background-position	%	*Положення фонового малюнка відносно лівого верхнього кута блоку*	BODY{ background-position: left top}
	Од. довжини		
	top		
	center		
	bottom		
	left		
	right		
Властивості списків CSS			
list-style-type		*Вид маркера*	
		Для маркованого списку	LI { list-style-type:circle}
	disc	Маркер	
	circle	Коло	
	square	Квадрат	
		Для нумерованого списку	LI { list-style-type: upper-alpha}
	decimal	Десяткові (арабські) цифри	
	decimal-leading-zero	Десяткові цифри з добавленням нуля спочатку	
	lower-roman	Малі латинські цифри	
	upper-roman	Великі латинські цифри	
	lower-alpha	Малі літери	
	upper-alpha	Прописні літери	
	none	Нумерація відсутня	
list-style-image	*Встановлює зображення в якості маркера*		LI { list-style-image: url(images/bullet.gif}
	URL	URL зображення	
	none	Зображення відсутнє	
list-style-position	outside	*Положення маркера відносно рядків тексту*	LI {list-style-position: inside};
	inside		
Властивості CSS для форматування границь і берегів			
PADDING	Одиниці довжини або відсотки до ширини блоку	*Відступи від границі елемента до його вмісту*	table { padding: 15px}
padding-top		Зверху	
padding-		Справа	

Властивість	Значення	Опис	Приклад
right			
padding-bottom		Знизу	
padding-left		Зліва	
padding		Однакового розміру для всіх сторін	
MARGIN	Одиниці довжини або відсотки до ширини блоку	*Розміри берегів*	p{ margin-top:30px; margin-bottom:5%}
margin-top		Зверху	
margin-right		Справа	
margin-bottom		Знизу	
margin-left		Зліва	
margin		Однакового розміру для всіх сторін	
BORDER		*Ширина рамки*	p{ border-top-width:3px}
border-top-width		Ширина рамки верхньої границі	
border-right-width		Ширина рамки правої границі	
border-bottom-width		Ширина рамки нижньої границі	
border-left-width		Ширина рамки лівої границі	
border-width		Ширина рамки всіх границь	
	thin	Тонка	
	medium	Середньої товщини	
	thick	товста	
	Одиниці довжини		
border-top-color	Значення кольору	*Колір рамки для границі*	p{ border-color:rgb(90,60,90)}
border-right-color			
border-bottom-color			

Власти-вість	Значення	Опис	Приклад
border-left-color			
border-color			
transparent			
border-style	none	*Стиль рамки*	table {border-style: double}
	dotted		
	dashed		
	solid		
	double		
	groove		
	ridge		
	inset		
	outset		
border-top border-right border-bottom border-left	Ширина, стиль, колір	*Визначає товщину, стиль і колір кожної границі*	table { border-top: solid 3px red; border-left: solid 3px green}
border		*Визначає товщину, стиль і колір всієї рамки*	table {border: solid 3px red}

6.3.9. Приклади застосування стилів на Web-сторінках

Оформлення за допомогою атрибута STYLE

Самий примітивний спосіб використання CSS – ввести в код HTML атрибут STYLE= з відповідним значенням. Його можна добавити практично до всіх тегів (крім таких, як, наприклад, <HEAD> і <HTML>).

Код, представлений в прикладі 1 можна змінити, використавши елементи стильового оформлення (рис. 6.8).

Атрибути тегів <FONT> та <DIV>були змінені, замість всіх використаний один атрибут STYLE=”….” з відповідними значеннями.

Звичайно, це не скоротило загальний код, оскільки всі параметри вводилися знову. Виграти тут можливо тільки за рахунок того, що атрибут STYLE надає більше можливостей для оформлення. Наприклад, тег DIV не може задавати абзацних відступів, і їх потрібно встановлювати за допомогою нерозривних пропусків. Стильова властивість text-indent визначає абзацний відступ в будь-яких одиницях. Проте, слід відмовитися від розподілу на абзацу за допомогою тегу <BR>, оскільки він не сприймає цей атрибут. Краще скористатися тегом <DIV> чи <P>:

<DIV STYLE="text-align: justify; text-indent: 2em;"> – визначений абзацний відступ, рівний двом символам максимальної ширини в даному шрифті.

```
<html> <head>
<title> Використання атрибуту Style
</title>
</head> <body>
<font style="font-family:Arial
Black;font-size:14pt; color:#0000ff;">
Особливості написання те-
гів.</font><br>
<div STYLE="text-align: justify;">
<font style="font-family:Times New
Roman;font-size:12pt;"> Комбінація з
відкриваючого і закриваючого тега
називається <font style="font-
family:Century Gothic"> контейнером
тегів. </font> "Вмістом тега" може
бути текст чи інші теги.</div>
<div STYLE="text-align: justify;">
Теги не чутливі до регістра. </div>
<div STYLE="text-align: justify;">
Теги не чутливі до їх розташування
на сторінці. </font>
</div>
</body></html>
```

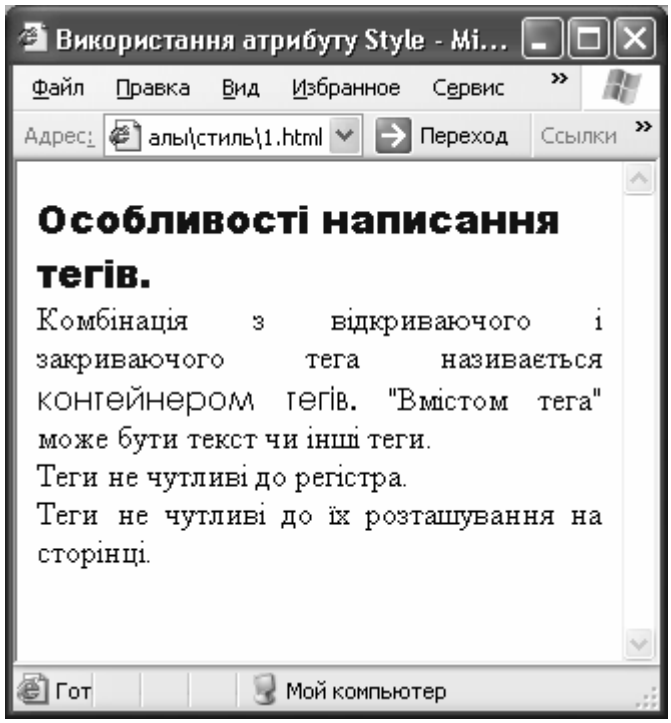


Рис. 6.8. Приклад застосування атрибуту STYLE="" для оформлення тексту

Визначення стилів в спеціальній таблиці

Атрибут STYLE= в основному використовується в тих випадках, коли потрібно призначити стиль до окремо взятого фрагменту. В інших випадках використовують так звані таблиці стилів.

Таблиця стилів частіше всього розміщується в заголовку HTML-документу, в розділі <HEAD>. Вона займає місце між тегами <STYLE> и </STYLE>.

Синтаксис таблиці стилів:

<STYLE> елемент, для якого визначається стиль{параметри стилю} </STYLE>

Пропуски, як і переноси рядків, тут не мають значення, важливо, щоб зручно і легко читався стиль. Адаптований програмний код попереднього прикладу та результат його перегляду в браузері представлено на рис. 6.9.

```
<html> <head>
<title> Використання таблиці стилів
</title>
<style>
div{text-align: justify;}
font.a1{font-family:Arial Black;font-
size:14pt; color:#0000ff;}
font.a2{font-family:Times New
Roman;font-size:12pt;}
font.a3{font-family:Century Gothic}
</style>
</head> <body>
<font class="a1"> Особливості напи-
сання тегів.</font><br>
<div>
<font class="a2"> Комбінація з від-
криваючого і закриваючого тега на-
зивається <font class="a3"> контей-
нером тегів. </font> "Вмістом тега"
може бути текст чи інші теги.</div>
<div>
Теги не чутливі до регістра. </div>
<div> Теги не чутливі до їх розташу-
вання на сторінці. </font>
</div>
</body></html>
```

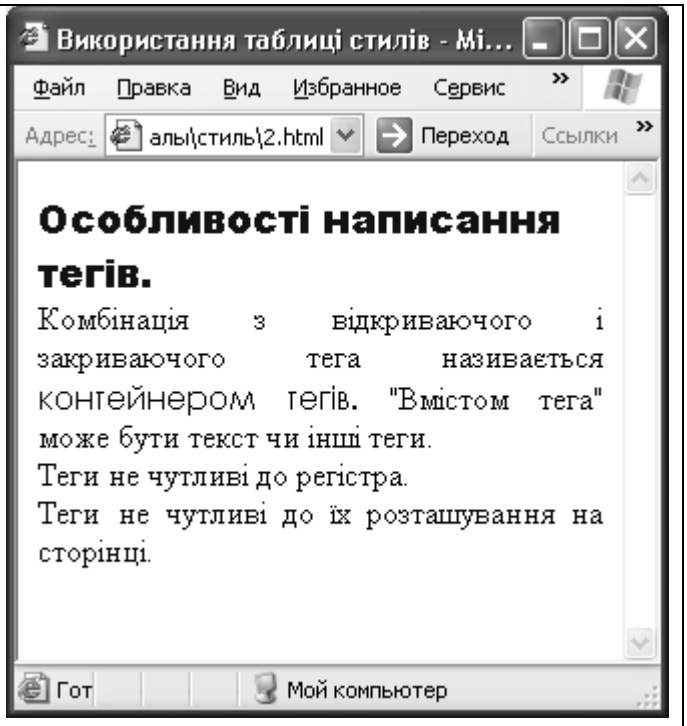


Рис. 6.9. Використання таблиці стилів

Результат виконання, як видно з рисунку, ідентичний. Значне скорочення коду буде досягнуто за рахунок опису параметрів тільки один раз. Вигодою є також однакове оформлення структурних блоків Web-сторінки: заголовків, основного тексту, визначень, зносок і т.і.

Зовнішні стильові таблиці

При розробці великих сайтів для збереження цілісності стиля, а також можливості швидко змінити який-небудь стиль у всіх файлах проекту, доцільно створювати таблиці стилів в окремому файлі.

Для цього слід написати всю таблицю стилів (без тегів <STYLE> и </STYLE>) в окремому файлі з розширенням .css, а потім “підключити” її у всі HTML- документи.

Наприклад, якщо файл таблиці стилів називається stt.css, то в розділ <HEAD> кожного HTML-документа, потрібно вставити наступний рядок:

<LINK REL="Stylesheet" HREF="stt.css" TYPE="text/css">

Другий спосіб “підключення” стильової таблиці – директива @import. Наприклад: <STYLE TYPE="text/css"> @import url (stt.css) ; </STYLE>

Оскільки ця директива не елемент мови HTML, вона повинна знаходитися між тегами <STYLE> і </STYLE>.

6.4. JavaScript

Розроблена в 1995 р. фірмою Netscape для версії 2.0 свого браузера мова JavaScript дотепер залишається допоміжною, але в той же час є абсолютно незамінним інструментом, що дозволяє завантаженій в браузер сторінці динамічно управляти своїм вмістом, а заразом і власне браузером. По своєму набору функцій ця мова близька до макромов, які з давніх пір вбудовуються в будь-яку достатньо складну програму або систему програм.

Як правило, для скриптів (script – сценарій) використовується дві основні мови програмування – JavaScript та VBScript.

JavaScript-сценарії вільно переплітаються і взаємодіють з HTML-розміткою сторінки. Вони дозволяють відкривати і закривати вікна браузера, завантажувати в них документи, управляти фреймами і взаємодіяти з полями форм (наприклад, перевіряючи правильність введених в них значень).

6.4.1. Розміщення скриптів

Сценарій, вбудований в документ за допомогою тега SCRIPT, може вставляти блоки HTML-коду в те місце документа, в якому розташований сам. Стала можливою плавна зміна кольору фону при завантаженні сторінки або «живі» меню, кожен пункт яких змінюється, коли над ним проводиш мишею.

Скрипти можуть розміщуватися в заголовку чи тілі HTML-документа. Їх місцезнаходження часто не впливає на роботу програми. Але ті, які повинні виконуватись в першу чергу, розміщують в заголовку.

Тег SCRIPT вставляється всередину тега HEAD у такому порядку:

```
<HEAD>
    <SCRIPT language="JavaScript">
        Команди мови JavaScript
    </SCRIPT>
<TITLE> Вміст заголовка </TITLE>
</HEAD>
```

При цьому, дія команд JavaScript розповсюджується на всю web-сторінку.

Зручно підключати зовнішні сценарії, написані на JavaScript.

```
<SCRIPT type=”text/javascript” src="http://server.com/progs/script.js”>
</SCRIPT>
```

В цьому випадку немає необхідності добавляти в контейнер <SCRIPT> вміст – на сторінку завантажиться сценарій script.js з вказаного місця. Ця можливість дуже зручна при використанні стандартних скриптів та цілих бібліотек.

Основні способи виконання скриптів – через виклик функції чи при настанні певної події (при завантаженні сторінки, натисканні кнопки і т.п.

6.4.2. Приклади застосування JavaScript

JavaScript має велику кількість команд і свій синтаксис. Наведемо декілька прикладів застосування цієї мови.

Функції затримки часу

Деколи в програмі потрібно створити певну затримку часу. Стандартна функція JavaScript setTimeout(); іноді не підходить, оскільки "на її фоні" можуть виконуватися інші функції, що може привести до небажаних накладок. Коректніший результат дає наступна функція:

```
function pause (mSec) { clock = new Date();
        justMinute = clock.getTime();
        while (true) { just = new Date();
        if (just.getTime() - justMinute > mSec) break; }
        }
```

Функції відкриття вікон

window.open("URL", "windowName", ["windowFeatures,..."]) де windowFeatures - це:

copyhistory [=yes/no] / [=1/0] – збереження історії завантаження документів у дане вікно;

directories [=yes/no] / [=1/0] – наявність в даному вікні кнопок груп новин;

height =pixelheight висота вікна в пікселях;

location [=yes/no] / [=1/0] наявність поля;

locationmenubar [=yes/no] / [=1/0] наявність меню;

resizable [=yes/no] / [=1/0] наявність рамки вікна, яка дозволяє змінювати його розміри;

scrollbars [=yes/no] / [=1/0] наявність лінійок прокрутки;

status [=yes/no] / [=1/0] наявність рядка стану;

toolbar [=yes/no] / [=1/0] наявність панелі інструментів;

width =pixelwidth ширина вікна в пікселях.

Використовувати пропуски в рядку windowFeatures не допускається.

Функції закриття вікон

Закрити вікно браузера можна за допомогою команди JavaScript window.close(). Для закриття поточного вікна можна використовувати префікс self: self.close(). Для того, щоб закрити певне вікно, потрібно звернутися до нього по імені: winName.close(). Проте слід відмітити, що подібний спосіб працює, тільки якщо вікно відкривалося за допомогою методу window.open(), причому значення, що повертається цим методом, було привласнене якій-небудь змінній:

```
win1 = window.open("http://www.webclub.ru",
            "winName", "height=60,width=175,scrollbars=no");
```

Якщо значення не привласнено, звернутися до вікна з батьківського неможливо. Потрібно згадати також, що спроба закрити останнє вікно браузера, зажадає підтвердження цієї операції користувачем. Така можливість була зроблена навмисно, щоб уникнути ситуації "хуліганського" закриття всіх вікон з боку JavaScript-програми.

Код посилання дня

Кожного дня тижня, на сторінку підставлятиметься новий URL. Він зручний для реклами своєї продукції або подачі новин.

```
<html>
<head>
<script language="JavaScript">
function GetTodaysURL()
{           var locationlist = new URLList
 (
      "1.html",       // Monday
      "2.html",       // Tuesday
      "3.html",
      "4.html",
      "5.html",
      "6.html",
      "7.html"        // Sunday       );
  now = new Date();
  num = now.getDay();
  if (num == 0) num = 7;
  location.href = locationlist.list[num-1];            }
function URLList ()
{           var argv = URLList.arguments;
  var argc = argv.length;
  this.list = new Object();
  for (var i = 0; i < argc; i++)
  this.list[i]= argv[i];
  this.count = argc;
  return this;       }
</script>
</head>
<body>
```

Приклад коду, що вибирає сторінку цього дня:

```
<a href="javascript:GetTodaysURL()">          </a>.
```

Код створення компактного меню

Всі посилання оформляються у вигляді елементів випадного меню, при натисненні на кнопку відбувається перехід на відповідну сторінку.

Ви напевно зустрічали компактні меню навігації, побудовані на основі випадного списку? Ось код скрипта, що дозволяє зробити таку навігацію:

```
<form name="f1">
<select name="Map" onChange="{
for (var i=0; i < this.length; i++) {
if (this.options[i].selected) {
if (i!=0) { top.window.location=this.options[i].value;break;}}}}">
<option selected value="#"> Швидкий перехід до розділів:
```

```
<option value="main.html">Газета
<option value="stat.html"> Статистика
<option value="price.html"> Ціни
<option value="repr.html"> Наші представники
<option value="reclam.html"> Приклади реклами
<option value="order.html"> Бюро замовлень
</select></select></form>
```

А якщо у вас використовуються фрейми, то достатньо буде змінити один рядок:

```
top.window.location=this.options[i].value;
```

на top.window.frames[ім'я фрейма в лапках або його номер].location=this.options[i].value;

Не забудьте, що нумерація фреймів в JavaScript починається з 0!

Код створення змінного привітання чи товару

Цей код дозволяє реагувати на час доби на стороні клієнта. Ви можете залежно від часу доби вивести відповідне вітання або запропонувати відповідний товар.

```
<html>
<head>
<script language="JavaScript">
function getHourOfDay()
{       var now = new Date();
       return(now.getHours());      }
function getTime()
{           var now = new Date();
       var minutes = now.getMinutes();
       var divider = ":";
       if (minutes<10)
              divider = ":0";
              // Коректне відображення часу відповідно версії 3.0
       if (navigator.appVersion.lastIndexOf('3.')
                     != -1 &&
              navigator.appName.lastIndexOf
                                   ('Netscape') != -1)
                  return( now.getHours()-1  + divider + minutes );
              // Інші версії повинні працювати з цим?
       return( now.getHours() + divider + minutes );          }
function sayHello ()
{                document.write( "Зараз <B>" +  getTime() + "</B>,  тому ми бажаємо
Вам" );  if(getHourOfDay()<5 || getHourOfDay()>19)
              document.write(' на добраніч!');
       else
       {    if ( getHourOfDay() < 11)
              {        document.write(' доброго ранку!');                    }
```

```
        else
        {       document.write(' хорошої роботи!');             }       } }
</script>
</head>
<body>
<script language="JavaScript">
<!--
sayHello()
// -->
</script>
</body>
</html>
```

Код лічильника відвідувань

Якщо Ви вважаєте, що Вашим відвідувачам важливо знати, скільки разів вони відвідали Вашу сторінку, скористайтеся цим кодом.

```
<html>
<head>
<script language="JavaScript">
// Вказана логічна змінна, якщо потрібно відображати сигнал, коли cookie –
        список фраз – перевищує 4KB, у такому разі вона = false
// name – ім’я cookie
// value – значення cookie
// [expires] – дата закінчення терміну cookie (визначення кінця поточної сесії)
// [path] – шлях, якщо cookie вірна (визначення шляху для викликання докумен-
        ту)
// [domain] – домен, якщо cookie вірна (визначення домену для виклику докуме-
        нту)
// [secure] – Логічне значення, що вказує, якщо передача cookie вимагає безпеч-
        ну передачу
// * значення аргументу за умовчанням
function setCookie(name, value, expires, path, domain, secure){
    var curCookie = name + "=" + escape(value)+
        ((expires) ? "; expires=" + expires.toGMTString() : "") +
        ((path) ? "; path=" + path : "") +
        ((domain) ? "; domain=" + domain : "") +
        ((secure) ? "; secure" : "")
    if (!caution || (name + "=" + escape(value)).length <= 4000)
        document.cookie = curCookie
    else
        if (confirm("Cookie exceeds 4KB and will be cut!"))
            document.cookie = curCookie     }
// name - name of the desired cookie
// * повертає рядок, що містить значення вказаного cookie або порожнього по-
кажчика, якщо cookie не існує
```

```
function getCookie(name)
{       var prefix = name + "="
        var cookieStartIndex = document.cookie.indexOf(prefix)
        if (cookieStartIndex == -1)
                return null
        var cookieEndIndex = document.cookie.indexOf(";", cookieStartIndex +
prefix.length)
        if (cookieEndIndex == -1)
                cookieEndIndex = document.cookie.length
        return unescape(document.cookie.substring(cookieStartIndex + prefix.length,
cookieEndIndex))          }
function deleteCookie(name, path, domain){
        if (getCookie(name)) {
                document.cookie = name + "=" +
                ((path) ? "; path=" + path : "") +
                ((domain) ? "; domain=" + domain : "") +
                "; expires=Thu, 01-Jan-70 00:00:01 GMT"              }   }
// date - який-небудь зразок об'єкту Date
// * ви маєте передати всі зразки об'єкту Date до цієї функції для налагодження
function fixDate(date){
        var base = new Date(0)
        var skew = base.getTime()
        if (skew > 0)
                date.setTime(date.getTime() - skew)    }
</script>
</head>
<body>
<script language="JavaScript">
<!--
var now = new Date()
fixDate(now)
now.setTime(now.getTime() + 365 * 24 * 60 * 60 * 1000)
var visits = getCookie("counter")
if (!visits)
        visits = 1
else
        visits = parseInt(visits)+ 1
setCookie("counter", visits, now)
document.write("Ви були тут " + visits + " раз(а).")
// -->
</script></body> </html>
```

6.5. Розробка форм на Web-сторінках

Якщо розглядати програмування на JavaScript в історичному ракурсі, то

першими об’єктами, для яких були розроблені методи і властивості, стали поля форм.

Форми використовуються в WWW для отримання відгуку користувача на надану інформацію та збору різноманітних даних про користувача. Після заповнення форми і запуску процесу її обробки інформація з неї потрапляє до програми, що працює на сервері. Простота використання тега <MAILTO:> в формах дозволяє навіть власникам невеликих сторінок отримувати відгук від своїх читачів. Для обробки великої кількості відгуків використовуються програми, що підтримують Common Gateway Interface (CGI). Таким чином користувач може інтерактивно спілкуватись з Web-сервером через Internet.

Прикладами форм являються запит на пошук інформації, форма для відкриття поштової скриньки, різноманітні анкети та опитування (рис. 6.10 та 6.11).

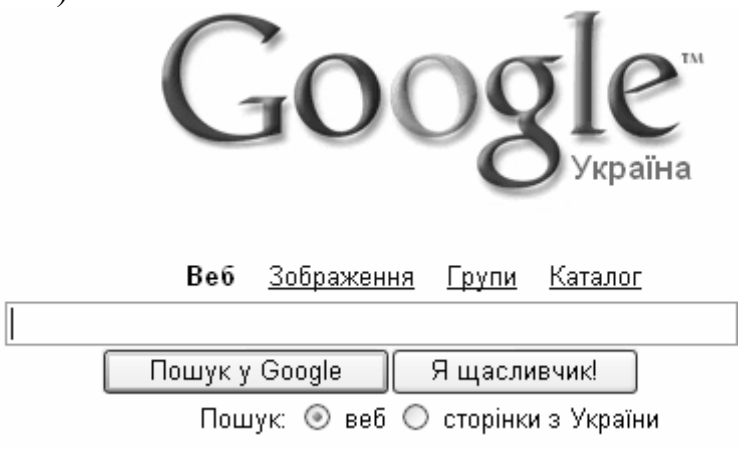


Рис. 6.10. Форма для введення параметрів пошуку

Рис. 6.11. Форма ідентифікації власника поштової скриньки

Можна відкрити запропоновані сторінки та знайти частину програмного коду, який відповідає за розміщення форми і обробку її даних на сервері.

6.5.1. Загальні правила створення та дизайну форм

Будь-яка форма містить:

1) Елементи форми, що являють собою стандартні поля для введення інформації;
2) Кнопку для відправлення даних форми на сервер;
3) Адресу програми на Web-сервері, що буде обробляти дані форми.

В HTML-документ форми поміщають за допомогою елементу-контейнера <form >. Він має наступний формат:

```
<form metod=”як відправляти” action="URL скрипта">
        елементи форми і зміст
</form>
```

Атрибут **method** визначає спосіб пересилання даних форми і може приймати значення post чи get. Post більш популярний, оскільки дозволяє пересилку великої кількості даних. Get встановлюється за замовченням, і використову-

ється для відправлення одиничних відповідей, що складаються з одного чи двох слів.

Атрибут **action** визначає URL скрипта, що буде приймати й обробляти дані форми. Часто скрипти розташовані на Web-сервері в каталозі bin/ чи cgi-bin/.

Одну форму не можна вкласти в іншу. Перед тим як створювати в документі нову форму, потрібно закрити попередню, інакше браузер проігнорує появу нової форми.

Перед програмною розробкою потрібно спроектувати її на папері і чітко уявляти її призначення та функції. Потрібно передбачити різноманітні можливості для спрощення заповнення форми користувачем. Він повинен вводити небагато інформації, але надати повну інформацію. Чим компактніша і зручніша форма, тим більше користувачів зголосяться її заповнити.

При розробці форми потрібно дотримуватися певних правил:

- Структурувати інформацію, відділяючі певні логічні блоки. Наприклад, в анкеті при реєстрації поштової скриньки на mail.ru (рис. 5.7) виділені блоки: введення логіну та паролю; параметрів секретного питання; додаткової інформації про користувача та ін. Блоки можуть відділятися лініями чи пропусками рядків.
- Виділяти за допомогою різноманітних ефектів форматування різні частини форми. Наприклад, на тій же формі анкети, коментарі та назви структурних блоків відформатовані інакше.
- Стандартизувати текст, тобто однотипні елементи (заголовки, зноски) потрібно і форматувати однаково.
- Створювати невеликі форми, щоб не злякати респондента великим обсягом питань і витратами часу, щоб на всі відповісти.
- Використовувати інтуїтивний дизайн. Якщо це можливо, використовувати радіокнопки (вибір статі), списки вибору (місяць народження, варіанти секретного питання, регіон та ін). Крім економії часу на заповнені форми, це суттєво полегшить обробку отриманих даних, оскільки виключає варіативність одних і тих же значень (регіон можна ввести Дніпропетровська область, Дніпропетровськ, Придніпров’я, та ін.), а також помилки при введенні.
- Попередньо перевірте форму для попередження можливих помилок. Можна спробувати заповнити її багато разів з введенням різних варіантів.

Для симетричного розміщення окремих елементів форми можна використовувати таблиці.

6.5.2. Елементи форми

Форма може мати один чи декілька елементів. Розглянемо основні з них більш детально.

Багаторядкове текстове поле

Використовується для отримання від відвідувача сайту повідомлень, на-

приклад відгуків, його думки про щось чи коментаріїв. Для добавлення текстового поля на форму використовується тег <textarea>. Без додаткових атрибутів створює порожнє (чи ні) текстове поле шириною 20 символів в два рядки (рис. 6.12).

Може мати наступні атрибути:

- name – ім'я-змінна, передається скрипту, що оброблює форму;
- cols – ширина рядку в символах, наприклад, cols="30";
- rows – кількість рядків, наприклад, rows="30";
- wrap може мати значення off (значення за замовченням) virtual (рядки тексту в формі розриваються автоматично, а на сервер відправляється суцільний текст) physical (при відправці на сервер, в місця вставки користувачем символів абзацу, добавляється розрив рядку).

```
<html> <head>
<title>Елемент Текстове поле</title>
</head>
<body>
<form>
<textarea> Тут Ви можете залишити Ваші коментарі
</textarea>
</form>
</body> </html>
```

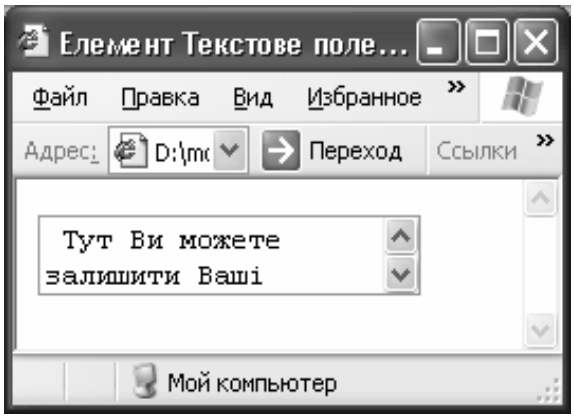


Рис. 6.12. Елемент Текстове поле

Елементи управління форми

До них відносять різноманітні кнопки, поля введення та ін.

Програмуються елементи управління за допомогою тегу <input>, який завжди використовується з атрибутами type та name.

Формат тегу <input>:

<input type="тип форми" name="ім'я-змінна" size="число" maxlength="число">

<input> одиночний елемент і закривати його, на відміну від <textarea>, непотрібно.

Одним із головних атрибутів є type, оскільки від його значення залежить тип елементу управління та значення інших атрибутів. Може приймати багато значень:

- text – створює текстове поле заданої довжини (рис. 6.13) в один рядок, причому довжина рядку і довжина тексту можуть відрізнятися.
- password – формує поле для введення паролю (рис. 6.13), введений текст захищає від підглядання заміною на зірочки, відправляє на сервер введені значення;
- checkbox – дозволяє користувачу вибрати декілька значень зі списку варіантів (рис. 6.14), застосовується коли є декілька правильних відповідей; за допомогою атрибуту cheked можна визначити один з елементів за замовченням, користувач може змінити цей вибір; атрибут

value визначає, що буде відправлено на сервер;

```
<form>
Введіть Ваше ім'я:
<input type="Text" name="login" size="20" maxlength="18">
<br>
<br>

Введіть пароль:
<input type="Password" name="passw" size="20">
</form>
```

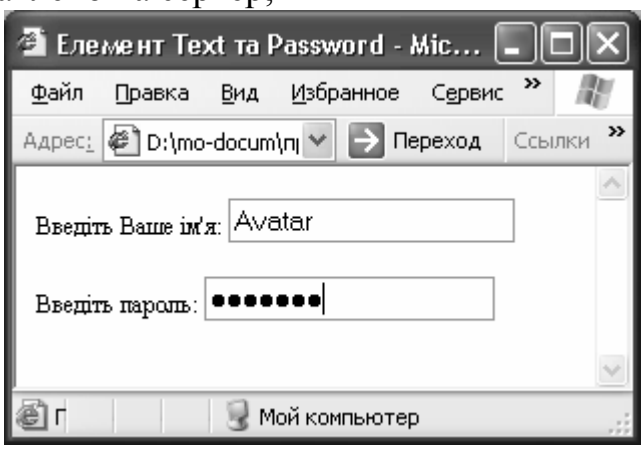


Рис. 6.13. Приклад програмування поля вводу тексту та паролю

```
<form>
Ваші захоплення:<br>
<input type="Checkbox" name="Sport" value="1"> Спорт<br>
<input type="Checkbox" name="Comp" value="2"> Коп'ютери<br>
<input type="Checkbox" name="Book" value="3"> Книги<br>
<input type="Checkbox" name="Cinema" value="4"> Кіно<br>
<input type="Checkbox" name="Music" value="5"> Музика<br>
</form>
```

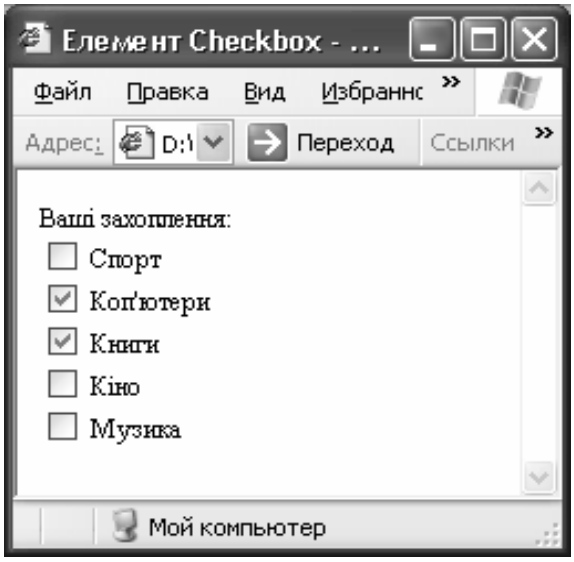


Рис. 6.14. Результат роботи елементу Checkbox

- radio – як і попередній, також пропонує вибрати варіант (рис. 6.15), але тільки один, для радіо перемикача атрибут name має однакове значення для всіх варіантів;

```
<form>
Ваша стать:<br>
<input type="Radio" name="sex" value="m">
Чоловіча
<input type="Radio" name="sex" value="f">
Жіноча
</form>
```

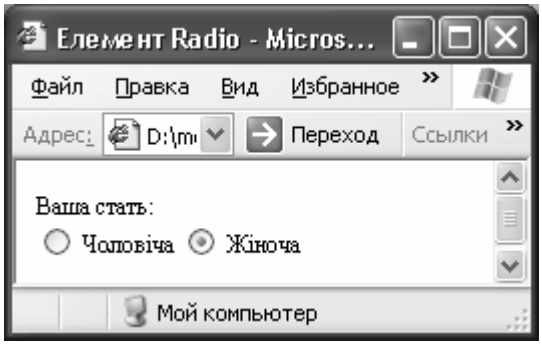


Рис. 6.15. Приклад використання перемикачів на формі

- hidden – використовується для передачі скрипту технічної інформації, яка може прискорити оброблення даних, на зовнішній вигляд фо-

рми не впливає;

- reset – кнопка для очищення вмісту форми (рис. 6.16), після натискання відновлюються значення, встановлені за замовченням, результат роботи кнопки на сервер не відправляється;
- submit – створює кнопку для підтвердження введення даних і відправки їх на сервер для обробки; атрибут value визначає підпис кнопки (рис. 6.16), розширити кнопку можна за допомогою пропусків;
- image – діє ідентично попередньому типу, але в якості кнопки використовується не текст, а графічне зображення (рис. 6.16), що застосовується для стилістичного оформлення сторінки;
- file – створює поле введення імені файлу (рис. 6.16), що відправляється на сервер, справа добавляється кнопка Обзор, яка відкриває вікно вибору з вмістом вінчестера.

```
<form>

<input type="Reset" value="Очистити форму"> <br><br>
<input type="Submit" value="Готово"> <br><br>
<input type="Submit" value="Відправити"> <br><br>

<input type="Submit" value=" ОК "> <br><br>
<input type="image" src="connected_networks.jpg" width="60" height="70"><br><br>
<input type="File">
</form>
```

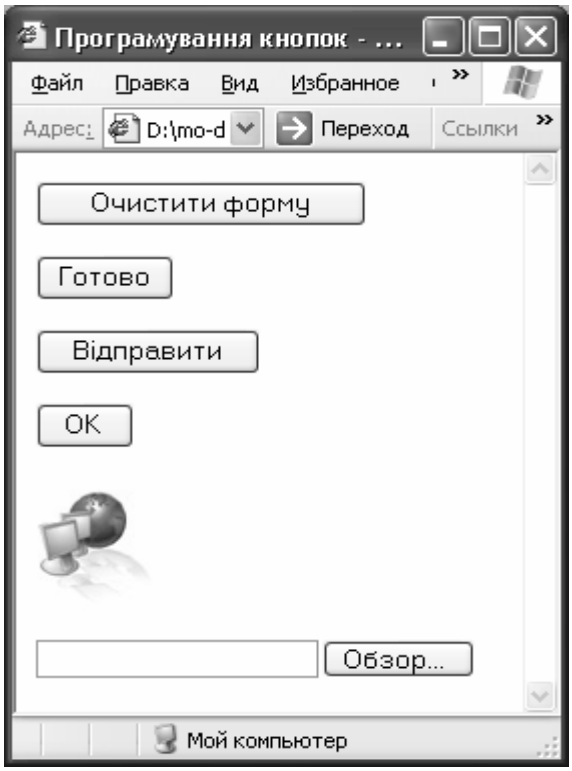


Рис. 6.16. Приклад поміщення кнопок

Випадаючі меню та списки, що прокручуються

Організуються за допомогою елемента <select>. Цей елемент також призначений для надання користувачу можливості вибору, текст вручну вводити не потрібно. Основними атрибутами елементу є name та size. Size визначає кількість опцій меню, що одночасно відображаються у вікні форми.

Формат елементу:

<select name=”Змінна”>

<option selected value=”значення”> Пункт меню

<option value=”значення”> Пункт меню

…

</select>

Формат елементу свідчить, що він є контейнером, тобто потребує обов’-

язкового закриття.

Приклад на рис 6.17 ілюструє можливості використання елементу.

Атрибут selected встановлює значення, що буде введене за замовченням.

Для перетворення випадаючого меню на простий список вибору встановлюються значення атрибуту size, змінивши значення на 5, отримаємо список вибору.

Можливість багатозначного вибору надає атрибут multiple, вибір здійснюється утримуючи кнопку CTRL і вибираючи із запропонованих елементів. Текст, виділений напівжирним в першому прикладі, повторюється в другому та третьому .

```
<form>
Звідки Ви дізнались про цей сайт?<br>
<select name="sel">
<option selected value="1"> За допомогою пошукової системи
<option value="2"> Знайдено в каталозі
<option value="3"> Випадково по гіперпосиланням
<option value="4"> Від друзів
<option value="5"> В спеціальній літературі</select>
<br><br><br>
<select name="sel" size="5">
….
</select>
<br><br><br>
<select name="sel" size="3" multiple>

</select>
</form>
```

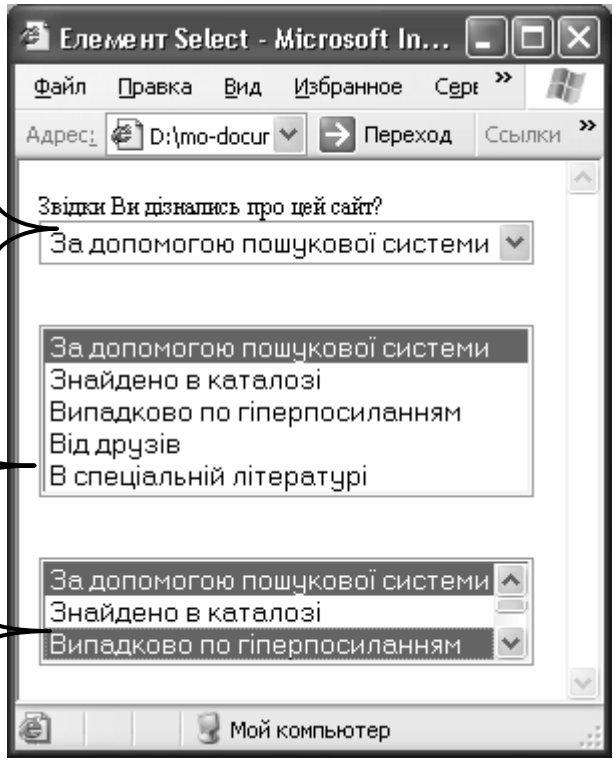


Рис. 6.17. Використання елементу select

6.5.3. Приклади складних форм

Приклад 1. Створення форми для підписки на розсилку

```
<html>
<head>      <title> Підписка на розсилку </title> </head>
<body>
<div align="center"><h1>Підписка на новини</h1></div>
<font size="+2"><div align="justify">Для отримання оперативної інформації з нашого сайту, заповніть і відправте реєстраційну форму.
Ваші дані будуть внесені до списку розсилки і вже сьогодні ви отримаєте повідомлення. </div>
<form method="post" action="http://www.vbnss.com/cgi-bin/subscribe">
```

```
Заповніть, будь-ласка форму:<br><br>
Ім'я: <input type="Text" name="First_n" size="20" maxlength="25"><br>
Прізвище: <input type="Text" name="last_n" size="20" maxlength="35"><br>
Логін: <input type="Text" name="login" size="20" maxlength="14"><br>
Введіть e-mail адресу <input type="Text" name="login" size="40" maxlength="40"><br><br>
Звідки Ви дізнались про цей сайт?<br>
<input type="Radio" name="know" value="1"> За допомогою пошукової системи
<input type="Radio" name="know" value="2"> Знайдено в каталозі
<input type="Radio" name="know" value="3"> Випадково по гіперпосиланням
<input type="Radio" name="know" value="4"> Від друзів
<input type="Radio" name="know" value="5"> В спеціальній літературі <br>
Виберіть періодичність розсилки<br>
<input type="Checkbox" name="Sport" value="1"> Щоденно<br>
<input type="Checkbox" name="Comp" value="2"> Раз на тиждень<br>
<input type="Reset" value="Очистити форму">
<input type="Submit" value="Відправити"> </font>
</form>
</body>
</html>
```

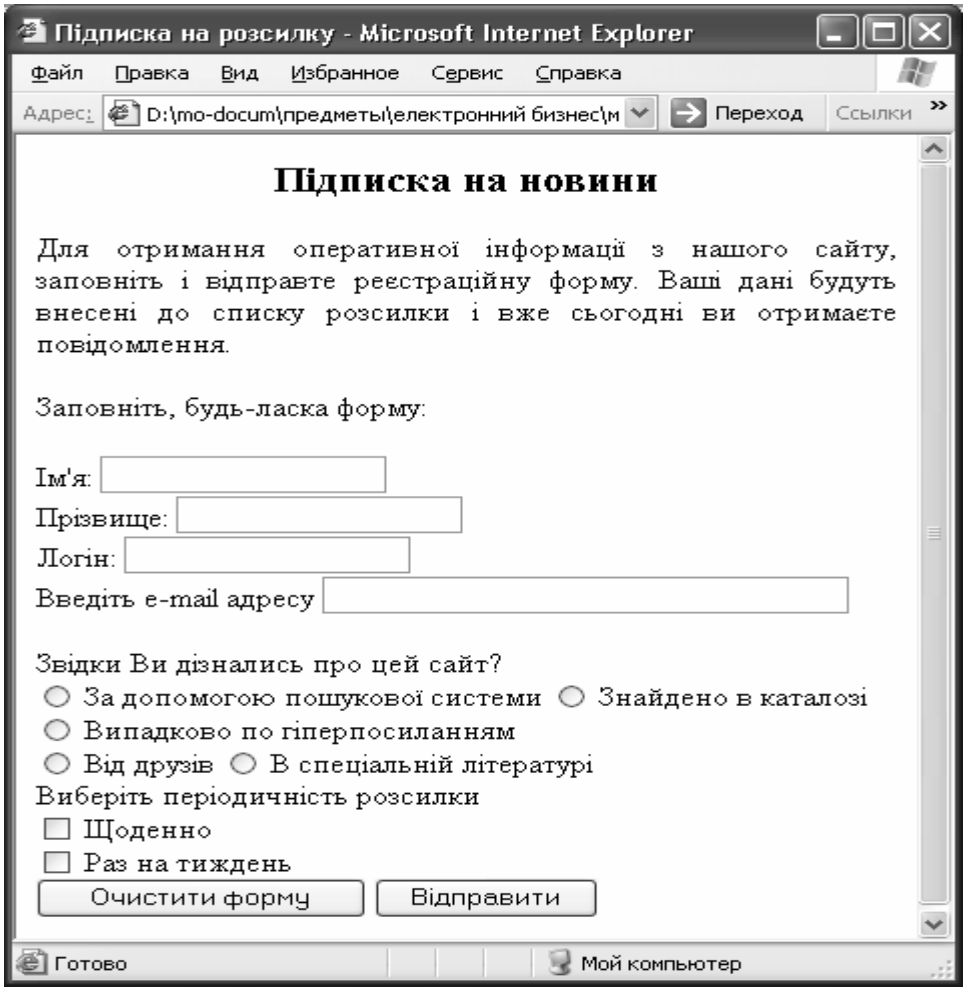


Рис. 6.18. Результат виконання програми по прикладу 1

Приклад

2. Створення форми вибору засобів оплати і доставки замовлення в Internet-крамниці

```
<html>
<head>      <title> Доставка та оплата </title> </head>
<body>
<font size="+2"><strong><div align="center">Вибір способу доставки та оплати</div></strong>
<form method="post" action="http://www.vbnss.com/cgi-bin/zakaz.pl">
<div align="justify">Обреріть спосіб доставки.</div>
<input type="Radio" name="DOST" value="K"> Доставка кур'єром (по Дніпропетровську)<br>
<input type="Radio" name="DOST" value="P"> Доставка поштою<br>
<input type="Radio" name="DOST" value="E"> Швидка Express-доставка<br>
<div align="justify">Обреріть спосіб оплати.</div>
<input type="Radio" name="OPL" value="Cash"> Оплата готівкою при доставці замовлення кур'єром<br>
<input type="Radio" name="OPL" value="bank"> Безготівковий розрахунок в українських гривнях<br>
<input type="Radio" name="OPL" value="Vms"> Оплата кредитними картками Visa чи MasterCard/EuroCard<br>
<input type="Radio" name="OPL" value="CM"> Оплата кредитними картками Cirrus/Maestro банку "Аваль"<br>
Якщо Ви маєте сертифікат на знижку, вкажіть його номер
<input type="Text" name="Name" size="20" maxlength="20">
<div align="center"><input type="Submit" value="Відправити"></div>
</form> </body></html>
```

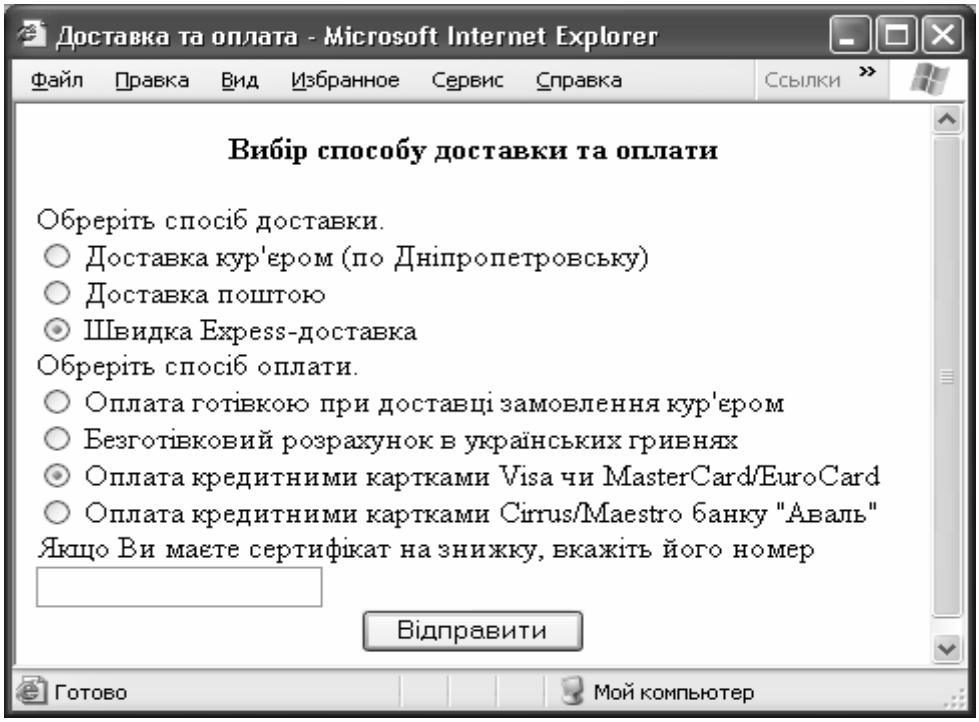


Рис. 6.19. Форма для прикладу 2

6.5.4. Обробка даних форми

Раніше вже відмічалось, що форми в Web-технології використовуються для обміну даними між клієнтами та сервером. Вони являються обов'язковим елементом сайтів електронної комерції.

Розглянемо більш детально питання розробки прикладного програмного забезпечення з боку серверу, тобто для обробки отриманих даних.

Для обробки великої кількості відгуків використовуються програми, що підтримують Common Gateway Interface (CGI), їх часто ще називають CGI-скрипти.

В основному CGI-BIN-скрипти призначені для отримання даних (змінних) від відвідувачів сайту і створення програмного HTML-кода («на льоту») у відповідь. Але можуть використовуватися для поміщення на сторінку лічильника відвідувань чи динамічних картинок.

Наприклад, для добавлення на Web-сторінку лічильника потрібно помістити в HTML-код команду:

<img src="cgi-bin/count.pl">

Це запустить скрипт counter.pl, що і поверне деяке значення змінної. Її значенням буде ім'я графічного файлу, що буде використано для відображення лічильника-малюнка.

Посилання на URL скрипта може бути розташоване де завгодно. У гіпертекстовому посиланні, можна використовувати адресу скрипта, що повертає випадкову Web-сторінку:

Цікаво, куди попаду?

Узагалі говорячи, написання скриптів трохи виходить за рамки цієї книги. Найчастіше CGi-BiN-скрипти пишуться на Perl, Ci, Visual Basic чи інших мовах. Якщо ви програміст, можна порекомендувати заглянути в книгу, у якій на серйозному рівні розглянуті всі питання CGI-програмування. Написання скриптів може бути дуже складною справою.

До найбільш популярних засобів розробки скриптів відносяться:

- shell (командна мова);
- Perl;
- C;
- Visual Basic.

Командні мови – перший інструмент програмування будь-якого досвідченого користувача. В Windows це cmd (Windows NT чи Windows 95), в Unix – shell різного роду.

Серед різних командних мов оболонок (shell) загальна для більшості Unix-платформ – GNU bash (Borne Again Shell). Для програмування CGI-скриптів bash зручна тому що наглядно демонструє різні властивості середовища Unix, які використовуються і в інших системах програмування. Крім того, часто програмування на командній мові використовується для порівняння програм, розроблених на інших платформах.

Крім cgi-bin-скриптів, для Web широко розповсюджена мова програмування — PHP. Це серверна мова сценаріїв, розроблена спеціально для Web. Розробка PHP була почата в 1994 році Расмусом Лердорфом. Спочатку PHP було

скроченням від Personal Home Page (Персональна домашня сторінка, з часом його назва була змінена на PHP Hypertext Preprocessor (Гіпертекстовий процесор PHP).

На відміну від cgi-bin-скриптів, PHP-код може бути поміщений в HTML. Крім того, це відкритий тобто безкоштовний ресурс, може вільно розповсюджуватися іншими користувачами чи організаціями. Домашня сторінка PHP знаходиться за адресою http://www.php.net. Поточну кількість доменів, що використовують PHP можна переглянути на http://www.php.net/usage.php.

Для накопичення даних, що надійшли від клієнтів використовуються спеціальні системи управління базами даних. PHP має вбудовані можливості підключення до багатьох СУБД, зокрема MySQL, PostgreeSQL, mSQL, Oracle, dbm, Hyperware, Informix, InterBase та Sybase.

Оскільки PHP був спеціально розроблений для використання у Web, він має багато вбудованих функцій для виконання різноманітних задач: на льоту генерувати GIF-зображення, підключатися до інших мережних служб, відправляти повідомлення електронної пошти, генерувати PDF-документи та ін.

Синтаксис мови успадкував багато рис C, Java, Perl, зручний для розуміння і використання.

Як і cgi-bin-скрипти, PHP-сценарії підключаються за допомогою атрибуту action.

Приклад. Потрібно зареєструвати користувача, що зайшов на сторінку. Форма буде мати наступний вигляд:

```
<form action="user.php" method="post">
<input type="text" name="name">
<input type="submit" name="send" value="ok">
</form>
```

Після введення відвідувачем імені і підтвердження вводу кнопкою ОК, він попадає на сторінку user.php (розширення .php показує, що у вказаному файлі повинні оброблятись команди PHP).

Приблизний сторінки user.php:

```
<html>
<head>
<title>Welcome!   </title>
</head>
 <body>
<! --звичайний html код-->
<?
        Код PHP
?>
 </body>
<html>
```

6.6. Програми для створення html-сторінок

Як вже зазначалося, документ, складений мовою розмітки HTML представ-

ляє собою текстовий файл, який можна створювати і редагувати в найпростіших текстових додатках наприклад в Блокноті чи редакторі WordPad.

Однак, нині існують більш розвинені і зручні програмні засоби. Їх умовно можна розділити на візуальні HTML-редактори і професійні редактори HTML-коду.

6.6.1. Візуальні редактори html-сторінок

У Web-редакторах типу WYSIWYG (What You See Is What You Get – „що бачите, те і отримаєте") користувач має справу не з кодом документа, а з графічними образами елементів HTML. Тобто створює не код, а оформлення сторінки, після чого програма автоматично підбирає для неї код. В результаті на сторінці часто виявляється багато зайвого коду. Він може залишатися, наприклад, від скасованих проб або коментарів, які програма вставляє сама, що безумовно сповільняє завантаження сторінки. До редакторів цього типу можна віднести: Microsoft FrontPage, Macromedia Dreamveawer, Netscape Composer, Star Office та ін.

Найпростішим способом створення web-сторінки є застосування текстового редактора Word. Спочатку засобами Word потрібно створити текст, включаючи, за необхідності малюнки, таблицю та різноманітні прийоми форматування. При цьому в головному меню по пункту „Вид" потрібно вибрати „Веб-документ". Тоді ваша сторінка виглядатиме так, як у браузері.

Якщо потрібно створити посилання на якийсь файл чи адресу, потрібно відмітити фрагмент тексту чи малюнок і вибрати пункт меню „Вставка–Гиперссылка" або натиснути кнопку  на панелі інструментів. Вікно, що відкриється (рис. 6.20) забезпечить вас можливістю вказати файл або адресу, на яку потрібно зробити перехід.

Після закінчення редагування потрібно обрати пункт меню „Сохранить как..." і вікні файл-менеджера, що відкриється обрати тип файлу – „Веб-страница".

При збереженні, Word самостійно згенерує HTML-код, згідно вказаного Вами форматування.

За допомогою Майстра Веб-сторінок можна створити навіть сайт з декількох сторінок.

Файли, створені таким чином часто мають надмірно великий обсяг і не завжди працюють так, як планувалося. Такий спосіб створення можна використовувати хіба що при створенні найпростіших персональних сторінок.

Зручніше скористатися можливостями браузера Opera. Якщо вибрати пункт меню "View-Source", то зображення web-сторінки зміниться на її текст, який можна редагувати так само, як Блокноті (рис. 6.21).

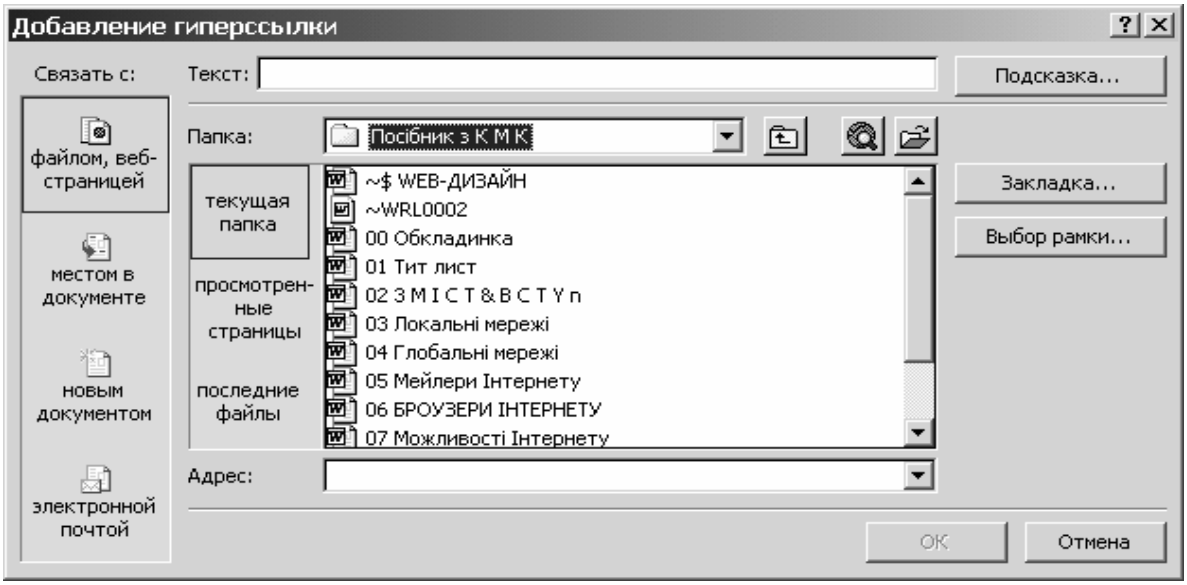


Рис. 6.20. Вікно додавання гіперпосилання у текстовому редакторі Word

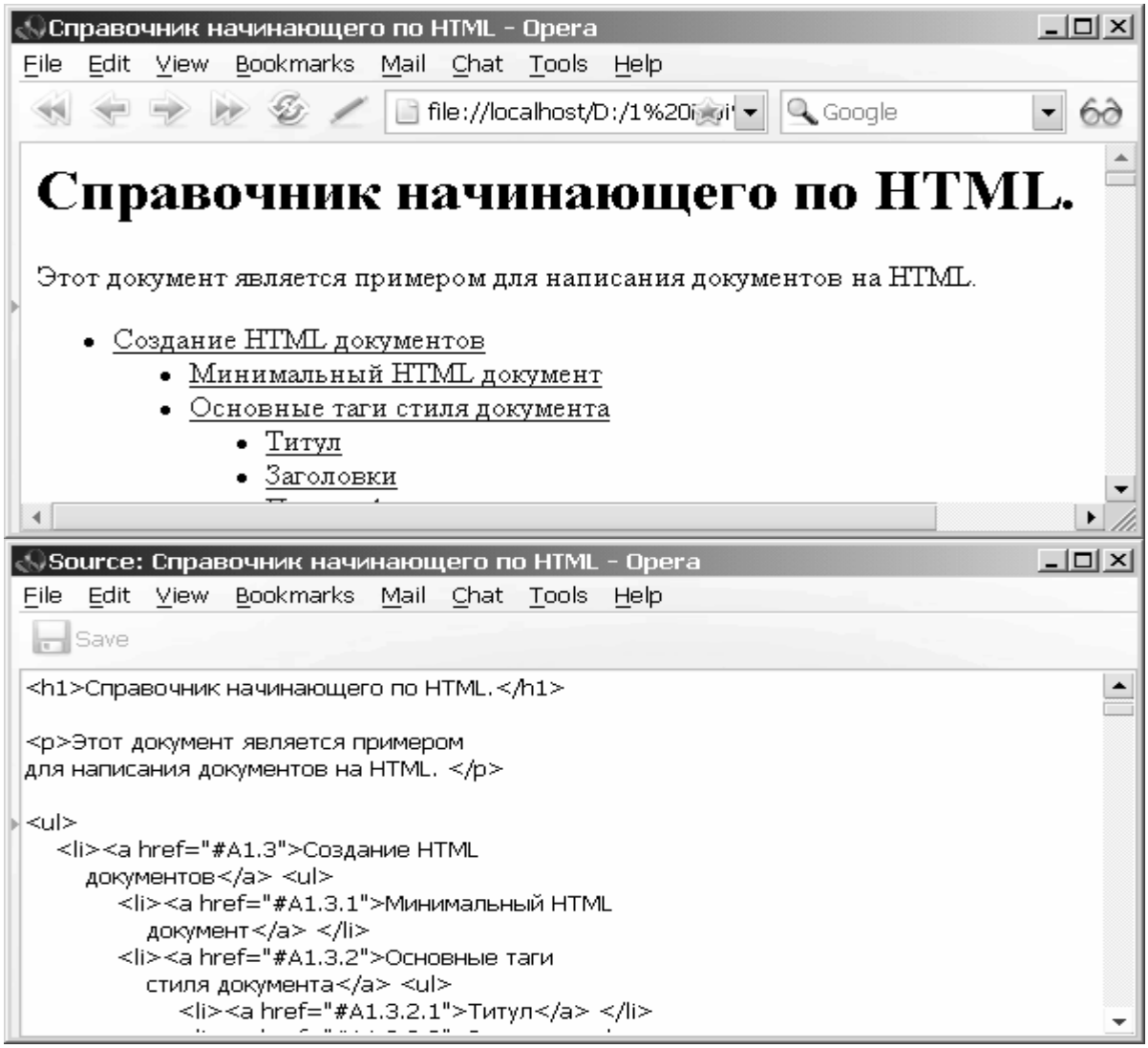


Рис. 6.21. Приклад редактора тегів броузера Opera

Повернення до зображення здійснюється через повторне відкриття того ж файлу, але перед цим треба зберегти зміни у тексті сторінки.

Розглянемо створення веб-сторінок за допомогою програми Page

Composer, яка є елементом NetScape Communicator`a. За структурою він дуже схожий на Word.

Меню ***Files/Publish*** призначене для відправки створеної вами html–сторінки на той сервер, який надає послуги по підтримці вашої сторінки в Interneti.

Про це треба домовлятися з провайдером. Є сервери, на яких ці послуги безкоштовні. Меню ***View/Page Sourse*** дозволяє побачити код, на якому написана сторінка, яку ви розробляєте.

Insert/Link дозволяє вставити гіперпосилання, так само, як і у Word`і на якусь Internet-адресу. ***Insert/Target*** дозволяє встановити зв'язок на інший елемент тієї самої сторінки. ***Insert/Image***, ***Horisontal Line, HTML Tag*** – це вставка відповідно малюнка, горизонтальної лінії та якоїсь HTML-тегу. ***Insert/New Line Break***, **Break Below Image** забезпечують вставку лінії розриву та обтікання текстом малюнка.

Починаючи створювати нову HTML-сторінку, треба вибрати її тип. Це буде чиста сторінка, сторінка з готових шаблонів, сторінка з майстра підказок чи сторінка з якогось файлу.

Треба мати на увазі, що шаблони та майстер підказок можуть бути відсутніми на вашому комп'ютері, тоді система зробить спробу зв'язатися через Internet з сервером компанії Netscape.

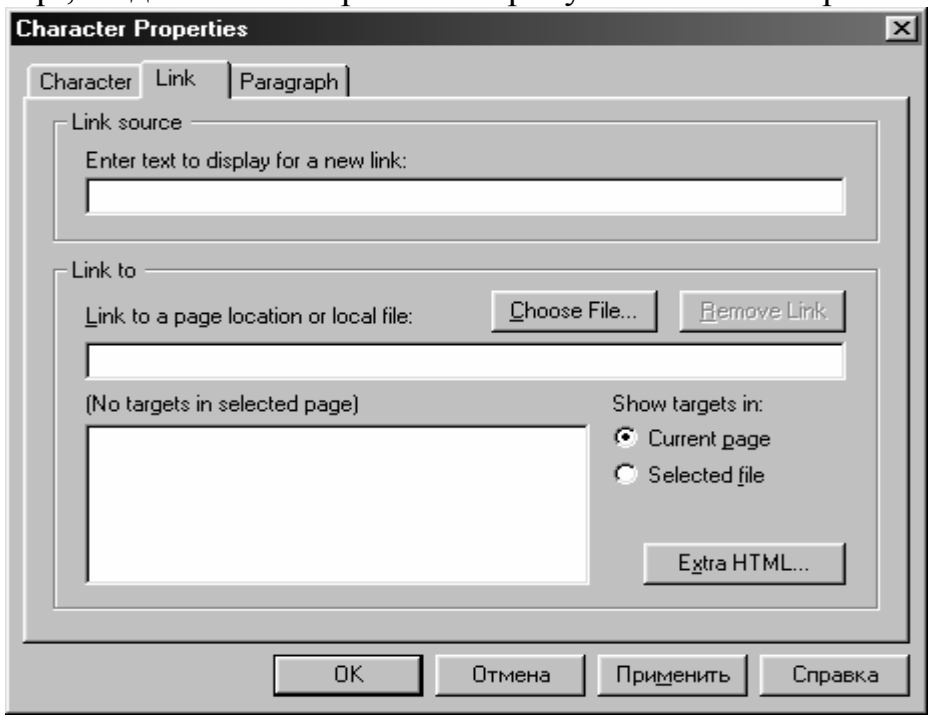


Рис. 6.22. Вікно створення гіперпосилання

Якщо ми додаємо таблицю, то вікно, що з'являється, схоже на подібне вікно у текстовому редакторі Word. Якщо створена таблиця нас не влаштовує, завжди можна правою кнопкою миші викликати контекстне меню в якому **Properties** означає властивості об'єкта (рис. 6.22), які можна змінювати.

Коли ми вставляємо зображення, з'явиться вікно, текст якого визначає всі параметри цього малюнку. Малюнки можна використовувати і як фон для тексту.

Далі, обов'язково треба вказати, де знаходитиметься файл з нашою сторінкою. Цю інформацію дає провайдер. Бажано доступ до сторінки закривати паролем, щоб ніхто її без вас не зміг змінити. Обов'язковим є вказування всіх файлів, які були приєднані до вашої сторінки (часто це малюнки), інакше в

Interneti їх не буде видно.

Існує також можливість створення веб-сторінок за допомогою Access та Excel. Для цього текст, який ви бажаєте зробити веб-сторінкою треба зберегти як html-файл.

Є ще один нескладний редактор веб-сторінок Microsoft FrontPage. Розберемо його детальніше.

Інтерфейс програми дуже схожий на структуру вікна редактора Word (рис. 6.23). Про Word нагадують панелі інструментів, головне меню, панель форматування та ін.

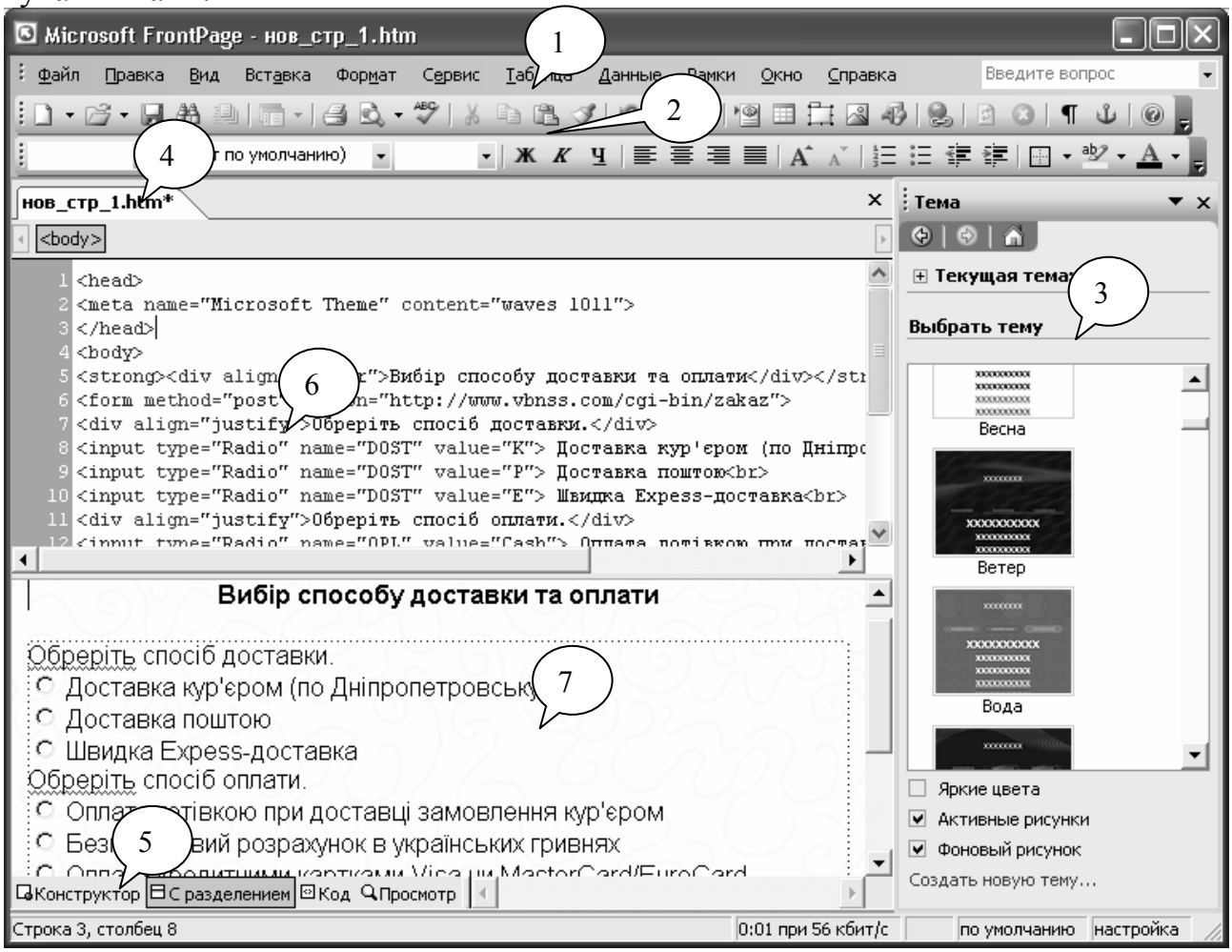


Рис. 6.23. Інтерфейс програми Microsoft Front Page

Основні елементи вікна:

1. Рядок головного меню.
2. Панелі інструментів. Є близько 10 стандартних, та можливість розробки власних.
3. Область задач. На вікні відкриті список запропонованих тем оформлення.
4. Панель закладок. Саме тут відображаються імена відкритих документів.
5. Меню представлень. Кожний документ можна відобразити в декількох виглядах:

- **Конструктор** – звичайний режим візуального оформлення сторінки, можливий не тільки перегляд, а й редагування;
- **С разделением** – вікно розділяється на дві частини, у верхній (6) виводиться код сторінки, в нижній (7) її зображення;
- **Код** – робота виключно з HTML-кодом;
- **Просмотр** – ілюструє сторінку у вікні браузера.

Меню **Правка** та **Вид** надають багато можливостей по оптимізації роботи та настроювання робочого вікна.

Редактор містить багато шаблонів графічного оформлення Web-сторінок, різноманітні приклади готових сайтів, колекцію розроблених інтерактивних елементів (лічильники, кнопки і т.п.), має засоби роботи з базами даних і інструменти для створення програмних анімацій.

Так само, як у документах Word, у Web-сторінках можна створювати маркіровані списки. Основна відмінність полягає в тому, що крім маркерів у списках можна використовувати графічні зображення.

Набір таких зображень знаходиться в діалоговому вікні **Список** (меню **Формат**). Крім графічних зображень, у діалоговому вікні пропонуються текстові маркери, підтримувані HTML для Web-сторінок. Для перегляду всього списку можливих графічних зображень натисніть кнопку **Другие**. Виберіть придатне зображення і натисніть кнопку **OK**, щоб повернутися в документ.

Графічні маркери зберігаються у форматі GIF (з розширенням .gif), у той же каталог, що і Web-сторінка, чи в каталог, зазначений щодо Web-сторінки. Якщо графічний маркер споконвічно є зображенням у форматі JPEG (з розширенням jpg), він зберігається у форматі JPEG.

Для зміни графічного маркера можна скористатися командою **Список**. Однак перш ніж змінити зображення, необхідно видалити всі наявні графічні маркери. Якщо при внесенні нових зображень старі не були вилучені, видаліть їх, виділивши і натиснувши клавішу DEL.

При створенні Web-сторінок не регулюються деякі параметри маркірованих списків, не підтримувані HTML. Наприклад, засоби редагування Web-сторінок не дозволяють змінювати відстань між тестом списку і маркером чи номером.

Нумерація списків на Web-сторінках майже не відрізняється від нумерації в документах Word. Відмінність полягає в тому, що для Web-сторінок неможлива автоматична нумерація структурних списків і заголовків. Однак щоб створити список з декількома рівнями, можна застосувати кілька стилів нумерації і відступів.

1. Виділіть текст, що є верхнім рівнем списку.
2. Виберіть команду **Список** у меню **Формат**, а потім — вкладку **Нумерованный**.
3. Виберіть потрібний формат номера, а потім натисніть кнопку **OK**. Щоб зрушити текст, що належить наступному рівню списку, установіть курсор на початку кожного абзацу чи рядка і натисніть клавішу TAB.

4. Хоча табуляція не застосовується в HTML, у редакторі символи табуляції списку перетворяться у відступи.

5. Виберіть команду **Список** у меню **Формат**, а потім — вкладку **Нумерованный**.

6. Виберіть формат номера для другого рівня.

7. Для кожного рівня в списку повторіть кроки 5-6.

На Web-вузлі Microsoft є додаткові маркери, фонові візерунки, горизонтальні лінії і шаблони.

Рисунки для Web-сторінок можна знайти на Web. При наявності доступу до Web, зображення можна завантажувати на свій комп'ютер.

1. Виберіть команду **Рисунок** у меню **Вставка**, а потім — команду **С Web-сторінки**.

2. Виконуйте вказівки, що будуть з'являтися на екрані.

Додаткові засоби оформлення Web-сторінок можна також знайти на Web. При наявності доступу до Web ці засоби можна завантажити на свій комп'ютер.

1. Виберіть команду **Создать** в меню **Файл,** а потім — вкладку **Web-страницы.**

2. Двічі клацніть **Дополнительные компоненты.**

3. Слідкуйте вказівкам, що будуть з'являтися на екрані.

До тексту інколи додають звуковий супровід, який може відтворюватися автоматично при відкритті Web-сторінки.

1. Виберіть команду **Фоновый звук** у меню **Вставка**, а потім — команду **Свойства**.

2. У поле **Звукозапись** введіть URL-адресу, якщо потрібно використовуйте кнопку **Обзор** для пошуку.

3. У поле **Число повторов** укажіть, скільки разів варто повторити запис. Для постійного звукового супроводу відкритої сторінки, виберіть **Незакончено**.

4. Щоб скопіювати звукозапис у ту ж папку, що і web-сторінку, установіть прапорець **Копировать в папку с документами.** Щоб використовувати відносний шлях, тобто шлях щодо поточної сторінки, установіть прапорець **Использовать относительный путь.**

Для прослуховування звукового супроводу користувачу необхідно мати звукову систему. Крім того, у його засобі перегляду Web повинна бути передбачена робота з файлами використовуваного формату запису звуку. У сторінку можна уставити файли звукозапису форматів WAV, MID, AU, AIF, RMI, SND і MP2 (MPEG audio). Звуковий супровід відтворюється автоматично при відкритті чи поверненні до сторінки.

Якщо сторінка відкривається часто, як, наприклад, основні сторінки, ці повтори можуть стати стомлюючими. Звуковий супровід можна переставити на сторінку, яку користувачі, імовірно, будуть відкривати не так часто. Іншим рішенням цієї проблеми є встановлення гіперпосилання, яке користувач може вибрати, щоб завантажити файл звукозапису:

1. Виділіть текст чи графічний об'єкт, що передбачається використовувати

як гіперпосилання, і натисніть кнопку **Добавить гиперссылку**.

2. Якщо в документі є не збережені зміни, відкриється вікно з пропозицією зберегти файл**.** Перед вставкою посилань рекомендується зберігати файл, особливо, якщо має бути створено відносне посилання, яке може стати у пригоді при переміщенні усіх файлів як групи.

3. Уведіть шлях до кінцевого файлу в поле **Связать с файлом/URL** чи натисніть кнопку **Обзор** і виберіть потрібний файл зі списку.

4. Якщо потрібно перейти до визначеного об'єкта файлу, укажіть цей об'єкт у поле **Имя обьекта** в документі**.**

Щоб перейти до визначеного об'єкта іншого файлу MS Office, вкажіть ім'я діапазону Microsoft Excel, ім'я об'єкта бази даних Microsoft Access, номер слайда Microsoft PowerPoint чи ім'я закладки Word, до якої потрібно перейти. Щоб вибрати потрібну закладку зі списку, натисніть кнопку **Обзор.**

Щоб швидко ввести адресу URL у поле **Связать с файлом/URL**, переконайтеся, що поле **Связать с файлом/URL** не заповнено, відкрийте засіб перегляду Web, відкрийте потрібну Web-сторінку і поверніться в Word. Після переходу по гіперпосиланню до об'єкта, документу чи сторінки відкривається панель інструментів Web. Натиснувши кнопку **Назад**, можна повернутися до початкової позиції в публікації Word чи на Web-сторінці.

На Web-сторінці можна розмістити також вбудований відеозапис. Для нього можна задати два варіанти відтворення: відразу після відкриття сторінки чи після щиглика сторінки мишею.

Оскільки не всі засоби перегляду Web передбачають можливість перегляду убудованого відеозапису, можливо буде корисним замінити її дублюючим текстом і зображеннями, чи взагалі не поміщати важливі дані у формі відеозапису. Рекомендується зберігати документи перед тим, як вставляти в них відеозапис.

Щоб зменшити розмір графічного файлу, використовуйте менше кольорів, зменшіть розмір чи кількість малюнків, там, де це можливо, використовуйте одинакові малюнки. Наприклад, якщо на всіх Web-сторінках для оформлення маркірованих списків використовуються однакові маркери, зображення маркера завантажується тільки один раз, навіть якщо він з'являється на декількох сторінках. Для редагування файлів зображення можна використовувати графічні редактори, наприклад Microsoft Photo Editor, що поставляється з Microsoft Office.

Для поділу тексту на частини на Web-сторінках часто використовуються горизонтальні лінії. Для їх вставки потрібно виконати наступні дії:

1. Клацніть те місце, де передбачається розмістити лінію.
2. Виберіть команду **Горизонтальная линия** в меню **Вставка.**
3. Зі списку **Вид** виберіть потрібну лінію або натисніть кнопку **Другие** і виберіть іншу лінію.

У Web-сторінку можна також уставити рядок, що біжить.

1. Виберіть команду **Бегущая строка**, у меню **Вставка.**
2. Уведіть текст рядка, що біжить, у поле **Ввести текст бегущей строки**.
3. Виберіть інші параметри.

При створенні Web-сторінки можна вказати мову, яку варто використову-

вати для перегляду цієї сторінки в засобі перегляду Web. Наприклад, щоб текст на сторінці зображувався грецькими буквами, необхідно установити грецьку мову. Можна задати мову, яка була використовувана за замовчуванням для кодування знову створюваних сторінок.

1. Виберіть команду в меню **Файл.**
2. У групі **Кодирование HTML** відзначте потрібні параметри.

Якщо сторінка зображена з застосуванням неправильного коду, укажіть мову, що має бути використана для перегляду сторінки, вибравши потрібну у списку для відображення цієї сторінки. Цей код буде використаний і для наступних сторінок, якщо код мови для них не може бути визначений.

Щоб задати кодування мови для збереження сторінки, виберіть потрібну мову в списку **Для сохранения этой страницы.**

Щоб задати кодування мови, яка використовується за замовчуванням для знову створюваних Web-сторінок, виберіть її в списку **Для создания новых Web-страниц (основное кодирование**).

Назва зображується в області назви засобу перегляду Web.

1. Виберіть команду **Свойства** в меню **Файл**.
2. У поле **Название** введіть потрібний заголовок.

Є функції, що дозволяють створювати Web-сторінки, не записуючи вихідні коди HTML. Однак при бажанні на створену сторінку можна вручну уставити власні коди HTML.

1. Уведіть вихідні коди HTML.
2. Виділіть їх.
3. Зі списку **Стиль** виберіть **Разметка HTML.**

FrontPage надає можливості по оптимізації Html-коду. Для цього досить в меню **Сервис** вибрати пункт **Оптимизировать Html-код** (рис. 6.24).

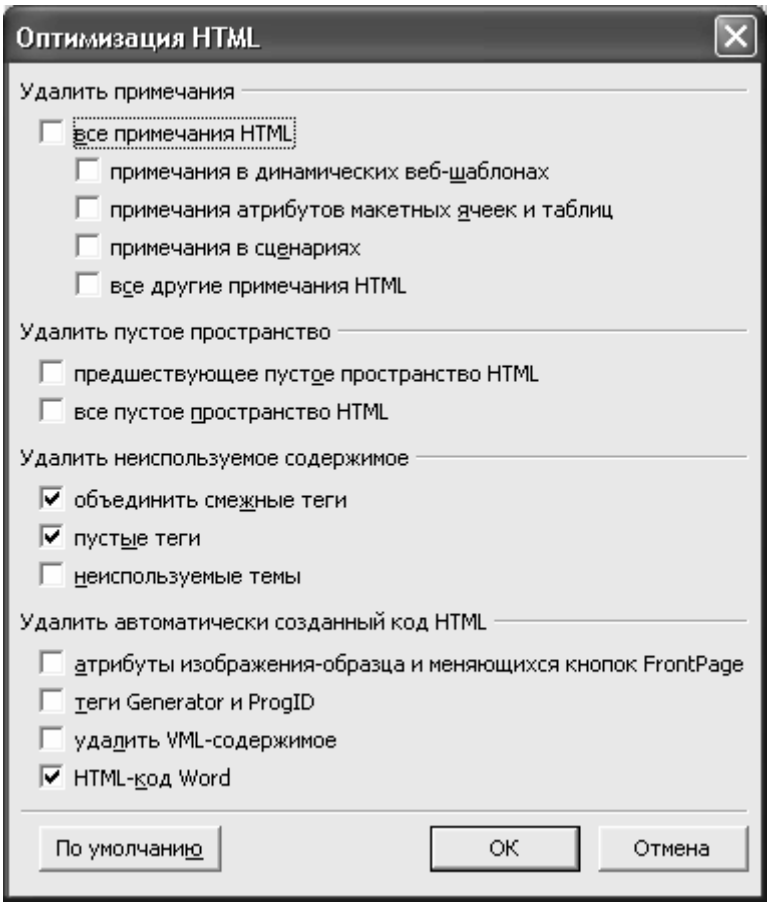


Рис. 6.24. Сервісні функції оптимізації коду

6.6.2. Редактори html-кодів

Серед редакторів html-текстів найбільш відомі Homesite та HotDog.

Використання цих програм передбачає знання користувачем мови HTML (а також, за бажанням, JavaScript і ін.). Вони автоматизують введення коду, пе-

ревіряють помилки і мають безліч доступних функцій, зручних для користувача. Тому їх можна розглядати не тільки як могутній інструмент розробки Web-публікацій професійної якості, але і як засіб навчання технології Web-дизайну.

До основних функціональних можливостей HomeSite 5 можна також віднести: швидкий перегляд сторінок у всіх браузерах, установлених на комп'ютері користувача; навігація по сторінках сайту, а також перегляд і навігація по ієрархії тегів усередині сторінки; перевірка коректності коду сторінки, а також контроль правопису текстового вмісту сторінок; автоматична перевірка коректності посилань у межах створюваного сайту; копіювання файлів проекту на віддалений сервер; підтримка технологій створення динамічних і інтерактивних сторінок; створення макросів і нових елементів інтерфейсу користувача засобами вбудованої мови програмування та ін.

Головне вікно HomeSite (рис. 6.25) надає користувачу, по суті, стандартний Windows-інтерфейс, що значно спрощує роботу з ним.

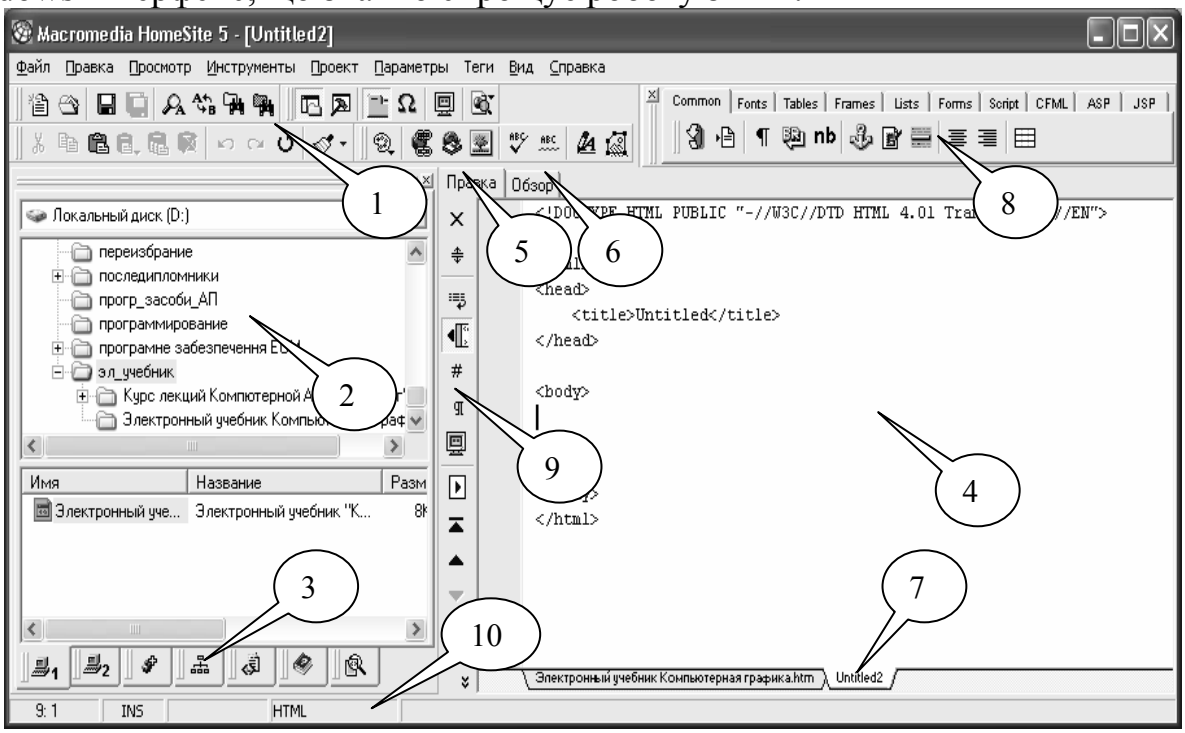


Рис. 6.25. Основні елементи інтерфейсу редактора HomeSite

Цифрами на рисунку показано наступні основні елементи:

1 – панелі інструментів, на них винесені найбільше часто застосовувані команди з усіх розділів меню, кожна кнопка має спливаючу підказку, управляти панелями можна за допомогою меню Вид;

2 – вікно ресурсів, назва досить умовна, оскільки в ньому може бути представлена найрізноманітніша інформація, наприклад, файлова структура проекту, перелік розділів довідкової системи, результати пошуку і т.д., при цьому кожен вид даних відображається на окремому "аркуші", переключення між якими здійснюється за допомогою ярликів ресурсів;

3 – ярлики ресурсів;

4 – вікно документа – у цій позиції екрана може бути представлене або вікно редактора коду (вкладка Правка – 5), вікно браузера (Обзор – 6, використо-

вується браузер, зазначений у параметрах настроювання HomeSite), чи вікно довідника (Help);

7 – ярлик документа – повідомляється ім'я файлу відкритого документа, кількість одночасно редагованих документів не обмежене;

8 – панель швидкої вставки тегів містить кнопки, що відповідають основним тегам мови HTML і елементам деяких інших мов, підтримуваних HomeSite, у залежності від типу зв'язаного з кнопкою тегу щиглик на ній приводить або до вставки тегу в текст документа, або до відкриття відповідного діалогового вікна редактора тегів, за допомогою якого здійснюється введення необхідних параметрів; для зручності роботи кнопки панелі згруповані по призначенню, переключення між групами виконується за допомогою вкладок груп;

9 – панель текстового редактора, дозволяє настроювати вид робочого вікна, нумерацію рядків, перевірку коректності тегів і т.п.;

10 – рядок стану виводить додаткову інформацію про виконувані операції.

Як і в інших Windows-додатках, перелік доступних елементів керування (зокрема, кнопок панелей інструментів) визначається поточним станом редактора і властивостями активного об'єкта. Наприклад, вкладка Help з'являється на екрані тільки після першого звертання до довідкової системи.

У залежності від типу файлу і виду виконуваної операції HomeSite надає користувачу різні засоби для роботи з ним: на основі команд і/чи на основі прямого маніпулювання (drag-and-drop -і "перетягни і залиши"). Наприклад, щоб зберегти на диску текстовий файл, що редагується, варто вибрати в меню File команду Save, а щоб створити посилання на графічний файл, можна перетягнути мишею його значок з вікна ресурсів у вікно документа.

Можливості HomeSite виходять за рамки "звичайного" текстового редактора. Оскільки сучасну Web-публікацію складно уявити собі без графіки (а також інших мультимедійних елементів), без підтримки інтерактивності й інших сучасних Web-технологій, то в HomeSite передбачена можливість роботи з даними, представленими в близько 50 різних форматах.

Для тих типів файлів, що підтримуються HomeSite, велика частина операцій може бути виконана за допомогою елементів інтерфейсу листа **Files** вікна ресурсів. Фактично це вікно забезпечує виконання тих же операцій при роботі з файлами, що і Windows Explorer і, крім того, виконання ряду специфічних операцій, зв'язаних з редагуванням Web-документів.

Деякі з цих операцій (такі як відкриття і збереження файлів, створення резервної копії перед заміною тексту у файлах) виконуються для поточної папки. Команди меню **File** відносяться до файлу, відкритому у вікні документа, а не до файлу, обраному у вікні ресурсів.

Цікавою властивістю редактора є також кольорова індикація елементів HTML-коду. Наприклад, коментарі – виділяються сірим, теги – темно-синім, атрибути –ярко синім, посилання – зеленим, текст – чорним і т.п.

Автоматична підстановка тегів, тобто, допомога пакету при введені тегу і підборі його атрибутів, дозволяє досить швидко виконувати розмітку та уникати помилок при наборі.

Одним з найважливіших достоїнств HomeSite є те, що в ньому реалізований дуже зручний і ефективний механізм роботи з каскадними таблицями стилів. Цей механізм базується на використанні самостійного додатка TopStyle Lite (виробник - фірма Bradbure Software, LLC). Він встановлюється разом з HomeSite при згоді користувача. Виклик цього додатка виконується безпосередньо з текстового редактора натисканням кнопки  (редактор стилів) або вибором аналогічного пункту в меню **Инструменты**.

Робоче вікно (рис. 6.26) дозволяє досить легко створювати оформлення, завдяки цьому стильове оформлення сторінок сайта може розглядатися як складова частина єдиного процесу редагування коду сторінок.

Оскільки в TopStyle Lite передбачена підтримка всіх існуючих на сьогоднішній день специфікацій мови CSS і найбільш розповсюджених браузерів, створене в HomeSite стильове оформлення вузла з високою імовірністю буде коректно відтворено браузерами потенційних відвідувачів вашого сайта.

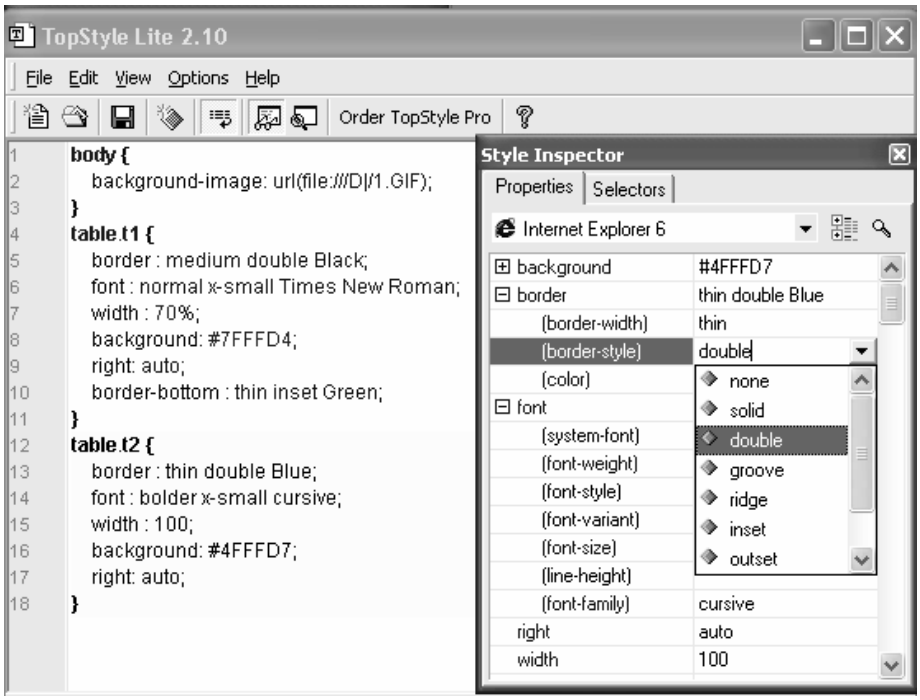


Рис. 6.26. Приклад створення стилю в редакторі TopStyle Lite

6.7. Публікація Web-вузла в Internet

Після закінчення створення HTML-документа, потрібно розмістити його в мережі Internet для забезпечення доступу до нього цільової групи користувачів. Попередньо потрібно виконати ще деяку підготовчу роботу:

1. Скопіювати всі файли в окрему папку і назвати її, наприклад, Website. У цій папці має бути файл index.html, що подає початкову сторінку сайту. Інші файли папки містять HTML-код інших сторінок (файли з розширенням *.html), також в даній папці розміщують файли каскадних таблиць стилів (з розширенням *.css). Імена усіх файлів повинні бути набрані малими латинськими букви-

ми. У папці сайту бажано передбачити папку для розміщення графічних елементів у форматі GIF чи JPEG (наприклад, IMG).

2. Перевірити роботу сайту, за необхідності внести зміни в HTML-код.

3. Вибрати Web-сервер, на якому цей сайт буде розміщено. Найбільш поширеною програмою Web-серверу є Apache Server, яка працює швидко і встановлюється безкоштовно. Часто використовуються також програми TomCat, Microsoft IIS, NCSA та ін. За допомогою них свій Web-сервер може створити будь-яка організації і навіть приватна особа. Однак, це не завжди виправдано, оскільки повноцінний Web-сервер повинен мати швидке з'єднання з Internet'ом (наприклад, через виділену лінію) і працювати цілодобово. Тому, багато організацій та приватних власників вирішують проблему Web-серверу за допомогою сторонніх організацій – провайдерів чи спеціалізованих фірм, які займаються хостингом.

Для того, щоб розмістити сайт в Interneti, необхідно отримати доступ на Web-сервер. Послуги по наданню місця для розміщення сайту, його обслуговуванню і технічному супроводу називаються хостингом (від англ. слова hosting спільне розміщення). Ресурси серверу і лінії зв'язку використовуються спільно безліччю клієнтів.

При виборі стороннього Web-серверу керуються двома основними критеріями: ціною і продуктивністю. Хостинг буває платний (від 85ум. од.) і безкоштовний (рис. 6.27).

Кількість безкоштовних серверів постійно росте і міняється, деякі час від часу перестають надавати такі послуги, наприклад www.narod.ru. До них безкоштовних можна віднести російськомовні сервери www.narod.ru, www.boom.ru, www.by.ru, www.webservis.ru, www.hotbox.ru. Такі сервери не вимагають оплати за підтримку Web-сайтів і існують за рахунок рекламодавців. Зареєстрованому клієнту надається обмежений, але достатній простір на диску серверу в середньому від 10 до 50 Мбайт. На ньому клієнт може розмістити як невеликий, так і солідний сайт. Єдине зобов'язання, що бере на себе клієнт безкоштовного Web-серверу, це розміщення на своєму сайті банерів – маленьких помітних зображень (часто анімованих), призначених для реклами.

Безкоштовний хостинг абсолютно не підходить для солідних проектів – сайтів великих фірм, відомих суспільних організацій, державних установ. Один тільки dns (тобто, доменна адреса), навіть без інших "принад" безкоштовного хосту, здатний серйозно зашкодити іміджу Вашої фірми.

Під продуктивністю роботи сервера мають на увазі наступні фактори:

- швидкість роботи серверу;
- обмеження стосовно типів і розмірів розміщуваних файлів;
- місце, що виділяється під сайт;
- наявність CSS;
- можливість запуску GGI та PHP скриптів;
- ненав'язливість реклами;
- „розкрученість" dns безкоштовного сервера та формат адреси Вашого сайту (narod.ru та jino-net.ru мають різні рівні популярності).

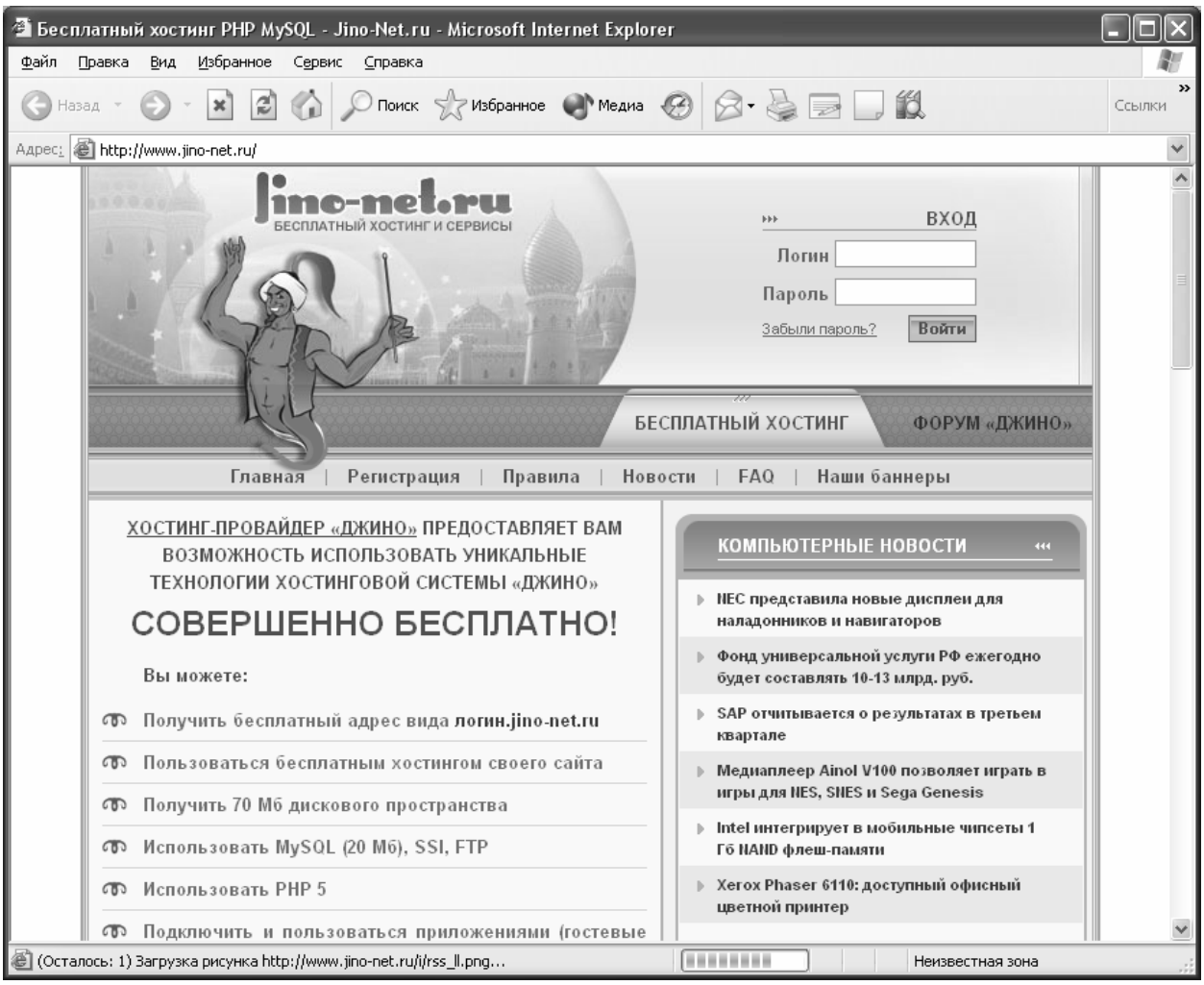


Рис. 6.27. Сервер безкоштовного хостинга Джино

При передачі файлів на безкоштовний сервер використовується протокол HTTP, тобто протокол, за допомогою якого здійснюється звичайний обмін файлами в WWW. Послідовність дій при розміщенні Web-сайту:

- підключитися до Internet і завантажити сторінку сервера (наприклад, www.boom.ru).
- Клацнути по посиланню *Регистрируйтесь и начинайте создание сайта* (чи подібне повідомлення). На наступній сторінці натиснути кнопку *Начать регистрацию.* після чого завантажиться сторінка для введення даних.
- Ввести ім’я свого сайту, наприклад, portal22 (адреса сайту матиме вигляд: www.portal22.boom.ru). Набрати пароль і підтвердити повторним набором. В поле E-mail ввести адресу електронної пошти. На цю адресу буде надісланий лист із підтвердженням реєстрації. В розділі *Дополнительная Информация о Пользователе* набрати своє прізвище, ім'я, нік. В нижній частині вікна набрати код для захисту від автоматичних реєстрацій. Після заповнення обов'язкових полів натиснути кнопку *Отправить.*
- Введені дані будуть відправлені на сервер. Ім'я сайту перевіряється з базою даних існуючих сайтів. Якщо воно унікальне, з'явиться новий діалог, у якому потрібно буде ще раз ввести ім’я сайту, пароль, E-mail та натис-

нути *Далее.* Якщо сайт з введеним іменем вже існує, потрібно буде повторити процедуру з більшою фантазією.

- Після успішної реєстрації буде виведена сторінка з вітанням, і зазначенням адреси сайту.
- Праворуч, у верхній частині сторінки вибрати пункт *Управление файлами,* після чого з'явиться вікно для завантаження файлів сайту Натиснути кнопку Browse (Огляд) і в стандартному діалозі знайти і вказати перший файл, для завантаження. Аналогічно за допомогою наступної другої кнопки Browse вказати другий файл і т.д. Після вибору всіх файлів, натиснути кнопку *Загрузить.*
- Після завантаження файлів з'явиться сторінка вмісту каталогу, на якій буде розмішений список файлів, скопійованих на сервер. Встановити прапорець напроти файла index.html і натиснути кнопку *Сделать главной.*

Для редагування вмісту папки сайту на сервері, можна користатися кнопками: *Создать*, *Редактировать*. *Копировать*. *Переместить*, *Переименовать*, *Удалить*. *Сделать главной*.

Для перевірки роботи створеного вузла, потрібно набрати його адресу у вікні браузера. переглянути його сторінки і роботу гіперпосилань.

Окремо слід зазначити, що на деяких хостингових серверах встановлені програмні засоби для розробки web-сторінок. Іншими словами, вам не потрібно мати спеціальне програмне забезпечення для створення html-документів. Звичайно, цією послугою в основному користуються для створення персональних сторінок.

6.8. Індивідуальні завдання №6

Всі ці завдання студенти виконують індивідуально. Оскільки, незважаючи на виконання однакових задач, всі елементи сайту мають нести індивідуальні риси його розробника.

6.8.1. Створення HTML-сторінок за допомогою Netscape Composer, Front Page, Word. Порівняння їх можливостей з особливостями роботи браузерів

Мета роботи: *Освоїти основні прийоми роботи по створенню HTML-сторінок за допомогою Netscape Composer, Front Page, Word.*

1. В редакторі Word за допомогою Майстра створити персональний сайт, з трьох сторінок:

а) перша містить загальну інформацію про Вас (Прізвище, ім'я, по батькові, дата народження, група навчання, і т.п.);

б) друга сторінка містить інформацію про Ваші захоплення та інтереси (хобі), бажано на сторінку вставити малюнок;

в) третя сторінка містить Ваші координати для зв'язку (адресу, телефон, E-mail).

2. У папці власної групи створити нову папку *САЙТ* і записати в неї всі файли створеного сайту.

3. Переглянути перелік створених майстром файлів та папок з малюнками.
4. Відкрити створений сайт за допомогою браузера.
5. Переглянути HTML-код сторінки. Віднайти основні структурні HTML-елементи сторінки.
6. Створити за допомогою Netscape Composer сторінку *Мої друзі.*
7. Вставити на першу сторінку сайту гіперпосилання на сторінку *Мої друзі* та на будь-який графічний файл та поштовий сервер.
8. Відредагувати першу сторінку за допомогою Front Page. Переглянути можливості по графічному оформленню сторінок.
9. Переглянути створений сайт і роботу гіперпосилань за допомогою браузера.

6.8.2. Особливості інтерфейсу Web-редактора Macromedia Homesite 5.0

Мета роботи: *Вивчити основні елементи інтерфейсу редактора та засоби автоматизації створення HTML-коду.*

1. Завантажити Web-редактор HomeSite 5.
2. Знайти основні структурні елементи робочого вікна.
3. Здійснити настроювання робочого вікна за допомогою меню *Вид.* Записати перелік команд цього меню.
4. Переглянути розділи довідки. До яких змін у вікні це привело?
5. Охарактеризувати набір тегів, що автоматично з'являються при створенні нового документа
6. Відкрити документи, створені на попередньому занятті.
7. Переглянути структуру сайту. Скопіювати структуру в текст звіту.
8. За допомогою панелі швидкої вставки тегів, розбити текстову частину сторінки на три рядки (тег <*BR*> вкладки **Common**).
9. Відкрити/закрити Вікно результатів.
10. Скористатися панеллю інструментів текстового редактора для зміни параметрів редагування документа.
11. Переглянути зроблені зміни за допомогою вкладки **Обзор**.
12. Закрити файли різними способами:
    а. через контекстове меню;
    б. за допомогою панелі інструментів текстового редактора;
    в. через меню Файл.

6.8.3. Проектування структури сайту та оформлення тексту на Web-сторінці

Мета роботи: *Навчитися проектувати структуру сайту, визначати доцільність системи навігації, координувати дії по підбору та структуруванню матеріалу, освоїти прості прийоми форматування тексту при формуванні рядків, абзаців, заголовків та створення списків в HTML-документі.*

Розробка структури сайту.

1. Спроектувати структуру сайту комерційної фірми, що надає послуги по розробці сайтів:
   - Підібрати матеріал (текст, малюнки);
   - Розбити матеріал по темам;
   - Визначити спосіб переходу між сторінками;
   - Вибрати структуру сайту;
   - Обґрунтувати доцільність вибраної структури.

Результат роботи представити у вигляді текстового файлу.

2. Використовуючи доступний вам редактор, почати розробку сайту, записуючи теги та їх вміст. Зберігати сторінки потрібно з розширенням *.HTML.

Переглядати отримані результати можна, застосовуючи будь-який браузер, через меню „Файл-Открыть” або “File-Open”. Якщо у файл-менеджері двічі клацнути лівою кнопкою мишки по такому файлу, то його вміст буде показано тим браузером, який визначено як системний.

3. Створити документ HTML наступного змісту і форматування:

Портал і К°

Фірма „Портал” лідер на ринку розробників професійних сайтів в Україні, надає консультаційні послуги по створенню, дизайну та супроводу сайтів. Фірма створена у **1996** році з метою впровадження web-технологій в бізнес-діяльність комерційних підприємств. *В 1998 році фірма стала власником домену .ks.ua.*

4. Фон сторінки – блідо-жовтий, колір тексту сторінки – сірий.
5. Назву оформити заголовком найвищого рівня та синім кольором.
6. Зберегти під іменем start.html.
7. Створити документ HTML наступного змісту:

Адреса:
м. Дніпропетровськ
вул. О. Гончара, 18 офіс 35
Фірма „Портал”
Телефон: (0562)48-17-28
E-mail: ip@portal.ua

8. Зберегти під іменем connect.html.
9. Створити документ HTML наступного змісту:

Фірма надає послуги:

1) Пошук інформації:
   - Надання цінової інформації:
   - Пошук конкуруючих структур;

- Аналіз інформації;

2) Розробка сайтів.
3) Розрахунок кошторису сайту.
4) Розміщення сайтів на власному порталі.
5) Супроводження сайтів.
6) Розміщення реклами в Internet:
   - Розробка банерної реклами;
   - Підбір сайтів для оптимального розміщення реклами.

10. Зберегти під іменем services.html.
11. Переглянути отриманий результат у вікні браузера.

6.8.4. Розміщення гіперпосилань та графічних елементів на Web-сторінках

Мета роботи: *Навчитися створювати внутрішні і зовнішні гіперспосилання на Web-сторінках. Засвоїти порядок розміщення малюнків на Web-сторінках.*

Ця робота є продовженням попередньої, тому всі створені раніше сторінки треба доповнити наступними даними:

1. Створити на сторінці start гіперпосилання на файли services і connect.
2. Створити на сторінці start перехід в кінець сторінки.
3. Створити додаткову сторінку „Наші проекти" де вказати гіперпосилання на наступні сервери:
   - Міністерство Аграрної політики України – http://www.minagro.kiev.ua
   - Сайт м. Дніпропетровськ – http://www.gorod.dp.ua
   - Сайт Дніпропетровського державного аграрного університету – http://www.dsau.dp.ua
   - Пошукова система Google – http://www.Google.com
   - Пошукова система Апорт – http://www.aport.ru
4. На створеній сторінці завантаження вузлів повинно проводитися в нове вікно браузеру.
5. Вставити логотип компанії на першу сторінку.
6. Створити буквицю і помістити в текст.
7. Вставити фрагмент WordArt. створений в текстовому редакторі MS Word.
8. Вставити малюнок зі сканера.
9. Вставити малюнок, створений в будь-якому графічному редакторі.
10. Зробити малюнок рухомим, щоби він з’являвся з будь-якої сторони сторінки.
11. Оцінити результат у вікні браузера.

6.8.5. Оформлення Web–сторінок з використанням стилів. Створення таблиць у Web-документах. Управління розміщенням елементів Web-сторінки за допомогою таблиць

Мета роботи: *Освоїти основні прийоми роботи створення таблиць на Web-сторінці, навчитися контролювати з їх допомогою розміщення інших елементів Web-сторінки.*

Ця робота є продовженням попередніх, тому всі створені раніше сторінки треба доповнювати наступними даними:

1. Створити стиль оформлення фону сторінки і записати в окремому файлі.
2. Підключити створений стиль до всіх сторінок сайту.
3. Задати для сторінок окремо стилі оформлення заголовків і абзаців, використовуючи класи.
4. Розробити стиль оформлення таблиці за допомогою редактора TopStyle, зберегти в окремому файлі.
5. Підключити стильову таблицю.
6. Створити таблицю, приведену нижче:

Вартість послуг, що надаються фірмою.

<table>
<tr><th rowspan="2">Вид робіт</th><th rowspan="2">Одиниця виміру</th><th rowspan="2">Розцінка, грн.</th><th colspan="2">Додаткові доплати</th></tr>
<tr><th>За термінове виконання</th><th>За обробку матеріалу</th></tr>
<tr><td>Пошук інформації:</td><td></td><td></td><td></td><td></td></tr>
<tr><td>• Надання цінової інформації:</td><td rowspan="3">Друк. арк.</td><td rowspan="3">1</td><td rowspan="3">10%</td><td rowspan="3">15%</td></tr>
<tr><td>• Пошук конкуруючих структур;</td></tr>
<tr><td>• Аналіз інформації:</td></tr>
<tr><td>Розробка сайтів.</td><td>Сторінка</td><td>20</td><td>10%</td><td>15%</td></tr>
<tr><td>Розрахунок кошторису сайту.</td><td></td><td>безкоштовно</td><td></td><td></td></tr>
</table>

7. Переглянути результати в браузері.

6.8.6. Динамічні елементи JavaScript на Web-сторінці

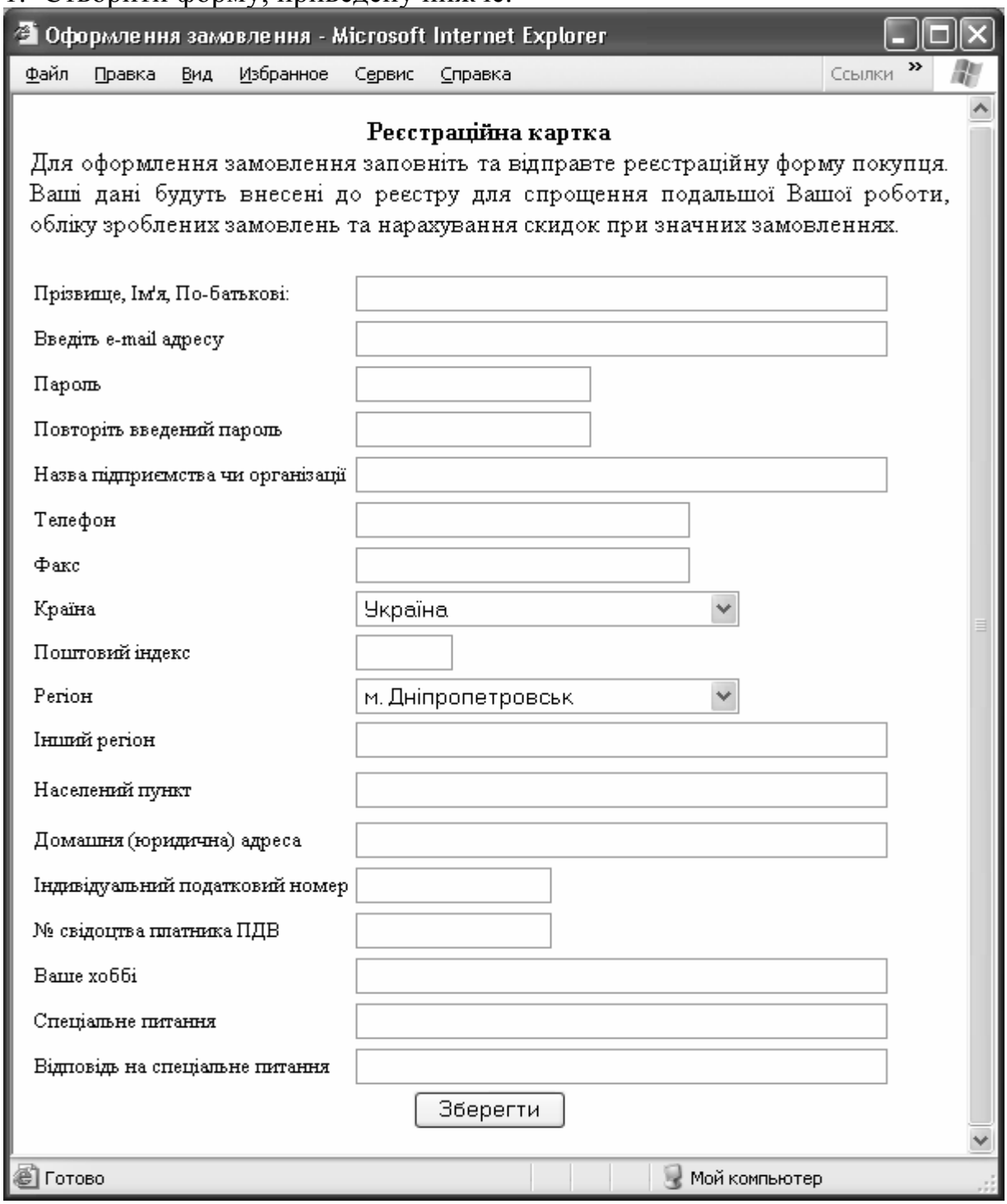
Мета роботи: *Засвоїти основні прийоми вставки скриптів на сторінки.*

1. Помістити на початкову сторінку сайту скрипт, який би змінював розміщене фото, в залежності від дня тижня.
2. Вставити в заголовок будь-якої сторінки розробленого сайту один із скриптів, запропонований в методичній частині.
3. Проглянути результат в браузері.
4. Перенести скрипт (створений по п.2) в тіло сторінки.
5. Переглянути в браузері та зробити висновки.

6.8.7. Розробка форм на Web-сторінках

Мета роботи: *Вивчити основні елементи розробки форм.*

1. Створити форму, приведену нижче.



2. Добавити на форму пункт вибору можливості підписки на розсилку новин.
3. Ввести код та порівняти отриманий результат.
4. Спроектувати форму для проведення маркетингового дослідження по вибраній темі. Використати всі основні елементи форми.
5. У звіті відобразити, які елементи форм використані і з якою метою.

6.8.8. Публікація Web-вузла в Internet

Мета роботи: *Освоїти основні прийоми по розміщенню сайту на безкоштовному Web-сервері за допомогою протоколу HTTP.*

1. Підготувати файли до розміщення в глобальній мережі.
2. За допомогою системи пошуку знайти 3 сервери, що пропонують безкоштовний хостинг.
3. Зробити порівняльну характеристику умов розміщення сайту на приведених серверах та оформити результати у звіті у вигляді таблиці.
4. Вибрати сервер для розміщення, переглянувши запропоновані умови.
5. Створити невеличкий сайт, вбудованими засобами створення web-сторінок. Темою сайту може бути: порівняльна характеристика хостингових сайтів.
6. Підготувати для розміщення в Internet сайт, створений в попередніх лабораторних роботах.
7. Зареєструвати власний сайт.
8. Скопіювати файли на сервер та перевірити роботу Web-вузла.

Контрольні запитання

1. HTML-сторінка і Web-сторінка це одне і те саме?
2. Дати коротку характеристику WYSIWYG-редакторам.
3. Що таке структура сайту?
4. Різниця між сайтами з лінійною і деревовидною структурою?
5. Дати приблизну класифікацію сайтів.
6. Описати способи створення Web-сторінки в редакторі Microsoft Word.
7. Перелічити і дати короткий опис основним елементам інтерфейсу HomeSite 5.
8. Дати характеристику редактору Microsoft FrontPage.
9. Які можливості по розробці Web-сторінок надає Opera?
10. Призначення мови HTML.
11. За допомогою яких тегів і атрибутів задаються параметри шрифту?
12. Як задати колір шрифту на сторінці?
13. Як змінити стиль нумерації в списку?
14. Якими атрибутами задаються товщина та колір рамок?
15. Назвіть теги, за допомогою яких створюються таблиці.
16. Напишіть HTML-код вставки таблиці 2x3.
17. За допомогою яких атрибутів виконується об'єднання клітинок таблиці?
18. Чи можна розбити клітинку на дві?
19. Як вирівняти таблицю по центру документа, по правому краю?
20. Які параметри таблиці задаються автоматично?
21. Що таке фрейм?
22. Які елементи Web-сторінок використовуються для розміщення її змісту?
23. Як задається розмір зображення?
24. Як задати ширину таблиці?

25. За допомогою якого елемента і атрибутів вставляється зображення на Web-сторінку?
26. Які формати графічних зображень використовуються в Internet і чому саме?
27. Які прийоми зменшення розміру зображення?
28. У чому полягає попередня робота зі створення Web-сайту?
29. На який сервер не можна помістити вашу HTML-сторінку?
30. Оцінити роль гіперпосилань в організації глобальної павутини World Wide Web.
31. Описати можливі атрибути тегу <A>.
32. Принцип роботи гіперпосилань.
33. Описати процедуру створення гіперпосилань.
34. Особливості стильового оформлення сайту цілком.
35. Що таке форма та які функції вона виконує?
36. Які основні елементи форм?
37. Як здійснюється обробка даних форм?
38. Перелічити основні функціональні можливості редактора HomeSite 5.
39. Порядок створення стильового оформлення для всієї сторінки.
40. Призначення CSS.
41. Які основні властивості CSS?
42. Що таке контекстові селектори?
43. Що таке успадкування?
44. Що розуміється під групуванням, і для чого воно використовується?
45. Описати функціональні можливості редактора стилів TopStyle.
46. Що таке скрипт?
47. Для чого створена мова JavaScript?
48. Правила створення скриптів.
49. Де на сторінці можуть бути поміщені скрипти?
50. Типи графічних форматів для Web-сторінок.
51. Типи стильового оформлення.
52. Що таке хостинг?
53. Які основні критерії вибору серверу для розміщення сайту?
54. У яких випадках доцільно для розміщення сайту користуватися послугами сторонньої організації?