

WEB-дизайн

Навчально-методичний посібник
Для студентів денної форми навчання
спеціальності «Інформатика»

WEB-дизайн

Навчально-методичний посібник
Для студентів денної форми навчання
спеціальності «Інформатика»,

УДК 004.738.52(075.8)

ББК 32.973я73

В26

Рекомендовано до друку рішенням редакційно-видавничої ради
СумДПУ ім. А.С. Макаренка

УКЛАДАЧ: кандидат філософських наук, доцент кафедри
інформатики СумДПУ ім. А.С. Макаренка Є.С. Голуб

РЕЦЕНЗЕНТИ: Шамоня В.Г., кандидат фізико-математичних наук,
доцент, завідувач кафедри інформатики СумДПУ
ім. А.С. Макаренка;
Алещенко С.С., викладач кафедри інформатики
СумДПУ ім. А.С. Макаренка

WEB-дизайн: Навчально-методичний посібник. Для
студентів денної форм навчання спеціальності «Інформатика». –
Суми: СумДПУ ім. А.С. Макаренка, 2008. – 70 с.

Посібник призначений для початкового ознайомлення з
основами створення HTML-документів за допомогою мови
гіпертекстової розмітки HTML 4.0. Містить велику кількість
прикладів побудови окремих елементів WEB-сторінки, завдання
для самостійної роботи та методичні вказівки щодо їх виконання.
Може бути використаний як довідник із теорії загального
сайтобудівництва.

УДК 004.738.52(075.8)

ББК 32.973я73

Зміст

Вступ	7
HTML і його подання	7
Елементи мови HTML	8
Атрибути тегів	9
Коментарі	10
Заголовок HTML-документа	10
Мета-теги	11
Завдання для самостійної роботи	12
Контрольні запитання	12
Тіло документа	12
Текстові блоки	14
Заголовки	14
Абзаци	14
Елемент-контейнер	15
Елемент-адреса	16
Елемент-цитата	16
Горизонтальна лінія	16
Вставка відформатованого тексту	17
Як зробити новий рядок?	17
Завдання для самостійної роботи	18
Контрольні запитання	19
Форматування шрифту	19
Завдання для самостійної роботи	20
Контрольні запитання	21
Списки	21
Пронумеровані списки	21
Марковані списки	21
Список визначень	22
Завдання для самостійної роботи	22
Контрольні запитання	23
Таблиці	23
Завдання для самостійної роботи	26
Контрольні запитання	27
Зображення	27
Завдання для самостійної роботи	30
Контрольні запитання	30
Гіперпосилання	31
Створення звичайного гіперпосилання	31
Графічний файл у ролі посилання	31

Посилання на e-mail	32
Закладки	32
Завдання для самостійної роботи	33
Контрольні запитання	33
Стилі	33
Завдання для самостійної роботи	45
Контрольні запитання	46
Селектори	46
Псевдокласи	47
Форми	47
Завдання для самостійної роботи	52
Контрольні запитання	52
Фрейми	52
Завдання для самостійної роботи	54
Контрольні запитання	55
Навігаційні карти	55
Спеціальні символи в HTML	59
Палітра кольорів	68
Література	70

Вступ

Основою всесвітньої «павутини» World Wide Web є мова гіпертекстової розмітки *HTML (Hyper Text Markup Language)*. Ця мова була створена вченими Європейського Центру Ядерних Досліджень (**CERN**, м. Женева). Наприкінці 80-х років у CERN зайнялися проблемою зберігання й відображення даних, які одержали колеги фізики. Складність полягала в тому, що кожний спеціаліст, який приїздив до ЦЕРНу, застосовував власні методи опису інформації. Тому було потрібно негайно створити універсальну систему, яка б не залежала від машинної платформи і водночас була б достатньо простою.

Ідея вирішення проблеми обміну документами між різними комп'ютерами полягала в тому, що документи повинні бути розмічені за допомогою певного коду – HTML. Такі документи могли б читатися на будь-якому комп'ютері, на якому встановлена всього лише одна програма перегляду – браузер.

Згодом HTML став основною мовою під час створення документів, розташовуваних у World Wide Web. Завдяки мові розмітки користувач Web може на своєму екрані переглянути документ у тому вигляді, у якому його задумав розроблювач: з певними розмірами шрифту й розбивкою на абзаци, із заданими розмірами й розташуванням малюнків й інших елементів.

HTML і його подання

HTML – набір угод для розмітки документів, які визначають зовнішній вигляд документів на екрані комп'ютера під час доступу до них з використанням програми браузера.

HTML-документ (або Web-сторінка) – це звичайний текстовий файл із розширенням *.htm або *.html, складений мовою HTML та змістовно призначений для публікації у WWW.

Код HTML є досить компактним, а HTML-документи мають розміри значно менші, ніж документи, підготовлені в текстових процесорах типу Word. Це одна з основних причин широкого застосування мови HTML для кодування інформації, розповсюджуваної в Інтернет.

HTML-документи розміщуються в WWW не поодинокі, а у вигляді сайтів.

Сайт – це сукупність Web-сторінок (кількох десятків, сотень або навіть тисяч), об'єднаних однією загальною темою й поміщених, як правило, на одному вузловому комп'ютері.

Елементи мови HTML

Документ, складений за допомогою мови розмітки HTML, є текстовий файл. Такий файл можна набрати й відредагувати у звичайному текстовому редакторі, наприклад, у додатках Блокнот або WordPad, які входять до складу ОС Windows.

Як виконується розмітка документа за допомогою HTML? Документ розбивається на елементи: заголовки, абзаци, малюнки, таблиці та ін.

Елемент HTML – це пара тегів і символічні дані (текст або код), розміщені між ними.

Для кожного елемента задається команда мови HTML, що називається **тегом (або дескриптором)**. У цій команді мыслиться інформація про те, як повинен виглядати даний елемент на Web-сторінці, які зв'язки він може мати з іншими елементами або документами.

Тег (з англ. tag – «показчик», «мітка») – це фрагмент коду, що описує певний елемент документа HTML і розташований у кутових дужках < >.

Найпростішим тегом є, наприклад, тег з ім'ям HTML. Тег <HTML> визначає початок HTML-документа. Початковому тегу <HTML> відповідає кінцевий тег з тим самим ім'ям, але з косою рисою </>, – це тег </HTML>. Таким чином, початок і кінець HTML-документа позначаються парою тегів:

```
<HTML>...</HTML>
```

Тут три крапки означають, що між початковим і кінцевим тегами може перебувати текст й (або) інші теги. У цьому випадку три крапки – це код усього документа.

Елемент звичайно називається за іменем тегу (без кутових дужок). Наприклад, елементом заголовка сторінки:

```
<TITLE> Простий приклад </TITLE>.
```

Усі елементи, передбачені у HTML, можна умовно розділити на чотири категорії:

1. **Структурні** – це елементи, які обов'язкові для документа, що відповідає стандарту HTML (наприклад, елементи HTML, HEAD, BODY й TITLE);
2. **Блокові** – це елементи, які призначені для форматування цілих текстових блоків (наприклад, елементи DIV, H1, H2 ,

H3, H4, H5, H6, P, PRE); часто блокові елементи відокремлюються перенесенням рядка від іншої частини документа;

3. **Текстові** – це елементи, які задають розмітку шрифту (I, B, U, BIG, SMALL й ін.), розмітку тексту (STRONG, CODE, VAR, CITE та ін.);

4. **Спеціальні** – це елементи порожнього рядка (BR, HR), якірний елемент A, упроваджені елементи (IMG, OBJECT, MAP та ін.), елементи таблиці (TABLE) та ін.

У будь-якому документі HTML обов'язково наявні **вкладені елементи**, тобто такі, які включені до складу інших елементів. Елемент TITLE вкладений в елемент HEAD, а елемент P – в елемент BODY. Елементи HEAD й BODY, у свою чергу, є вкладеними в елемент HTML.

Слід відзначити, що під час запису елементів HTML можна використовувати як прописні, так і малі літери. Так, браузер однаково буде сприймати теги <TITLE>, <Title> або <title>. Однак для однакової імена тегів записують прописними буквами.

HTML-документ умовно можна розділити на три частини:

1. Службова інформація для браузера що розташована у рядку з тегом <!DOCTYPE>.

2. Заголовна частина документа у якій міститься назва документа, може розташовуватися службова інформація для серверів, опис невеликих програм-сценаріїв. Ця частина документа укладена між тегами <HEAD> й </HEAD>.

3. Тіло документа що міститься між тегами <BODY> . . . </BODY> (іноді замість тегів <BODY> можуть використовуватися теги фреймової структури <FRAMESET>).

Атрибути тегів

Часто теги, крім імені, містять додаткові елементи, які називаються атрибутами. Наприклад, якщо в тег тіла документа <BODY> увести додатковий елемент: <BODY bgcolor="yellow">, то це буде означати, що документ повинен відображатися на жовтому фоні. Слово bgcolor є атрибутом, а yellow – значенням атрибута.

Атрибут – це компонент тегу, що містить указівки про те, як браузер повинен сприйняти й обробити тег. Атрибут записується після імені тегу перед закриваючою кутовою дужкою і складається, як правило, з пари «ім'я атрибута = значення».

Значення атрибута записується після імені атрибута через знак рівності (=). Усі значення атрибутів за замовчуванням повинні

бути розміщені в подвійних (") або одинарних (') лапках (іноді браузери розуміють запис без лапок). Імена атрибутів можуть набиратися як рядковими, так і прописними буквами. Браузер буде інтерпретувати їх однаково. У тегу може бути кілька атрибутів, у цьому випадку вони відокремлюються один від одного пробілами.

Коментарі

Часто під час створення Web-сторінок виникає необхідність пояснити певну особливість HTML-коду або наміри розроблювача, але так, щоб ці пояснення не відображалися у вікні браузера. У таких випадках незамінними є коментарі.

Вставка коментарів: <!--... --> використовується для створення коментарів у будь-якій частині документа. Усе, що перебуває усередині <!--... -->, елемент або текст, буде ігноруватися браузером (не буде оброблятися й виводитися на екран).

! Коментарі, розташовані всередині елемента TITLE (як, зрештою, і будь-які інші теги) трактуються як текст і будуть видні в заголовку документа. Іншими словами, тег <!-- -> усередині TITLE не діє.

! Не лінуйтеся коментувати все, що написали. Це чарівне правило заощадить вам чимало часу, коли ви намагатиметеся розібратися у власному утворі пізніше.

Заголовок HTML-документа

Заголовок створюється за допомогою елемента HEAD, між тегами якого розміщуються елементи, що містять технічну інформацію про документ. Тут повинна бути інформація про кодування сторінки, у нашому випадку: <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">, і теги, що визначають назву сторінки. <TITLE></TITLE> визначає ім'я всього документа. Ім'я, як правило, відображається в заголовку вікна браузера. Цей елемент **обов'язковий** для будь-якого HTML-документа й може бути зазначений не більше одного разу.

Мета-теги

Мета-теги – інформація для браузера й пошукових машин. У HTML-документі вони завжди розміщуються між тегами <HEAD></HEAD>. Більшість мета-тегів є необов'язковими, тому вирішуйте самі, які використовувати.

1. Кодування (обов'язковий мета-тег). Завдяки ньому браузер буде правильно відображати текст сторінки.

```
<meta http-equiv="content-type" content="text/html; charset=Кодування">
```

Приклад:

```
<meta http-equiv="content-type" content="text/html; charset=Windows-1251">
```

2. Показ додаткового файлу перед основним. Ви можете використати цей мета-тег для переадресації, не застосовуючи скриптів.

```
<meta name="Refresh" Content="число (у секундах); Url=ім'я файлу з розширенням">
```

Приклад:

```
<meta name="Refresh" Content="10; Url=INDEX.HTM">
```

Цей приклад завантажує документ INDEX.HTM через 10 секунд.

3. Указівка імені автора. Впишіть своє ім'я або Nickname.

```
<meta name="Author" Content="ім'я автора">
```

Приклад:

```
<meta name="Author" Content="Смірнов Іван">
```

4. Опис документа. Текст, що Ви вкажете у цьому тегу, буде відображатися пошуковими машинами. Довжина опису не повинна перевищувати 200 символів.

```
<meta name="Description" Content="опис">
```

Приклад:

```
<meta name="Description" Content="Усе про злом програмного забезпечення. Інструменти, віруси, програмні коди.">
```

5. Ключові слова для розвідувачів. Укажіть пошуковим машинам, згідно з якими словами їм здійснювати пошук інформації, що міститься на Вашій сторінці. Не пишіть у цьому теги однакових слів. Довжина списку – до 800 символів.

```
<meta name="Keywords" Content="слова через кому або пробіл">
```

Приклад:

```
<meta name="Keywords" Content="crack cracking злом">
```

6. Указівка e-mail.

```
<meta name="Reply-to" Content="адреса e-mail">
```

Приклад:

```
<meta name="Reply-to" Content="Lenin@INC.net">
```

7. Дата створення сайту.

```
<meta Name="Date" Content="місяць, число, рік і час через пробіл">
```

Приклад:

```
<meta Name="Data" Content="May 28 1999 15:34 Am">
```

Завдання для самостійної роботи

- У текстовому редакторі Блокнот уведіть такий код:

```
<HTML>
<HEAD> <TITLE>Заголовок документа</ TITLE >
</HEAD>
      <BODY>
      Зміст документа
      </BODY>
</HTML>
```
- Додайте рядок декларації версії HTML **!DOCTYPE**.
- Додайте мета-теги, що вказують кодування, ім'я автора й ключові слова для пошукових машин.
- Перегляньте як відображається даний документ в Internet Explorer.

Контрольні запитання

- Яка структура HTML-документа?
- Взаємозв'язок тегів й елементів у HTML-документі?
- Які існують категорії елементів?
- Яка функція атрибутів тегу?
- Для чого потрібні коментарі в документі?
- Як працює коментар з тегом <TITLE>?

Тіло документа

Тіло документа створюється за допомогою елемента BODY. Саме в тілі документа міститься все те, що ми бачимо на сторінці. А на сторінці ми звичайно бачимо: блоки тексту (причому тексту форматowanego), гіперпосилання, різноманітні списки, таблиці, усілякі об'єкти, картинки, заповнювані форми.

Теги <BODY>...</BODY> вказують початок і кінець тіла HTML-документа. Між початковим і кінцевим тегами розміщуються всі HTML-елементи, відповідальні за відображення документа,

керування ним та гіпертекстові посилання. Елемент BODY повинен зустрічатися в документі не більше одного разу.

Атрибути:

MARGINHEIGHT – визначає ширину (у пікселях) верхнього й нижнього полів документа (працює тільки в браузерях подібних Netscape);

TOPMARGIN – визначає ширину (у пікселях) верхнього й нижнього полів документа (працює тільки в браузерях Internet Explorer);

MARGINWIDTH – визначає ширину (у пікселях) лівого й правого полів документа (працює тільки в браузерях подібних Netscape);

LEFTMARGIN – визначає ширину (у пікселях) лівого й правого полів документа (працює тільки в браузерях Internet Explorer);

BACKGROUND – визначає зображення для "заливання" фону. Значення задається у вигляді повного URL або імені файла з картинкою у форматі GIF або JPG;

BGCOLOR – визначає кольори фону документа;

TEXT – визначає кольори тексту в документі;

LINK – визначає кольори гіперпосилань у документі;

ALINK – визначає кольори підсвічування гіперпосилань у момент натискання;

VLINK – визначає кольори гіперпосилань на документи, які ви вже переглянули;

Значення атрибутів BGCOLOR, TEXT, LINK, ALINK й VLINK задаються або RGB-значенням у шістнадцятиричний системі, або одним з 16 базових кольорів.

Приклад:

Створити елемент BODY, указуючи такі атрибути: фонову картинку (довільну), колір фону – зелений, колір тексту – чорний, колір посилання – чорний, колір посилання під час звертання до нього – червоний, колір посилання після звертання – синій, відступи зверху/знизу – 30 пікс., ліворуч/праворуч – 40 пікс.

<HTML>

<BODY BACKGROUND="images*.jpg" BGCOLOR="green"

TEXT="black" LINK="black" VLINK="red" ALINK="blue"

MARGINHEIGHT="30" TOPMARGIN="30" LEFTMARGIN="40" MARGINWIDTH="40">

... Текст документа ...

</BODY>

</HTML>

! Задаючи ширину полів для забезпечення сумісності з усіма видами браузерів використовуйте одночасно атрибути MARGINWIDTH/MARGINHEIGHT й TOPMARGIN/LEFTMARGIN як показано в прикладі.

! Завжди вказуйте атрибути BGCOLOR й TEXT одночасно. Вважайте, що вони неподільні. Якщо один із цих атрибутів не зазначений, браузер за замовчуванням буде використовувати кольори з поточної колірної схеми Windows. А це не завжди правильно відображується на машинах користувачів.

Текстові блоки

У цьому розділі описані елементи, що розбивають текст документа на блоки тим певним способом. Типовими прикладами текстових блоків є параграфи, абзаци й глави. Для відділення однієї частини тексту від іншої також використовуються розділові горизонтальні лінії й символи повернення каретки.

Заголовки

Заголовки поділяють документ на логічно закінчені блоки. Завдяки заголовкам відвідувач Web-сторінки може одержати інформацію про зміст. Щоб задати заголовки використовуються парні теги <H1> (заголовок 1-го рівня), <H2> (заголовок 2-го рівня) тощо. Усього передбачено 6 рівнів заголовків.

Браузер відображає заголовки більшим (напівжирним) шрифтом. Чим вище рівень заголовка, тим крупніше шрифт. Заголовки відокремлюються порожніми рядками від іншого тексту.

Приклад:

Задайте заголовок 5-го рівня: <H5>Заголовок</H5>

Абзаци

Під час створення Web-сторінок потрібно прагнути, щоб тексти були лаконічними й допускали чіткий розподіл на абзаци.

Вміст HTML-елементів звичайно записується з відступами. Це не відбивається на зовнішньому вигляді самої Web-сторінки, але полегшує читання коду.

Для створення нового абзацу використовується парний тег <P>, а для переходу на новий рядок без створення абзацу – тег
. Ці теги закривати не обов'язково. Звичайно, якщо Ви не використовуєте в тегу <P> атрибут ALIGN, яким може задаватися вирівнювання абзацу:

<P ALIGN=LEFT> По лівому краю </P>

<P ALIGN=CENTER> По центру </P>

<P ALIGN=RIGHT> По правому краю </P>

<P ALIGN=JUSTIFY> Текст, що перебуває між цими елементами вирівнюється по ширині </P>

 – цей елемент здійснює перенесення рядка, аналогічне з натисканням Enter у текстовому редакторі. Після того, як у браузерах з'явилася можливість обтікання зображення текстом (див. атрибут ALIGN елемента IMG), знадобився додатковий атрибут CLEAR. Елемент не має кінцевого тегу.

Атрибути:

CLEAR – указує на необхідність завершення обтікання зображення текстом. Може приймати такі значення:

all – завершити обтікання зображення текстом;

left – завершити обтікання текстом зображення, вирівняного по лівому краю;

right – завершити обтікання текстом зображення, вирівняного по правому краю;

**! Візьміть собі за правило: завжди ставити
 після тегу . У протилежному разі всі картини будуть мати неприємний відступ.**

Елемент-контейнер

<DIV> – використовується для логічного виділення блоку HTML-документа. У сучасному сайтобудівництві використовується як зручний контейнер для об'єктів сторінки, яким легко динамічно маніпулювати – переміщувати, вмикати/вимикати, створювати шари, регулювати відступи тощо. Текст або HTML-елементи, що знаходяться в цьому елементі, за замовчуванням оформлюються як окремий параграф.

Атрибути:

ALIGN – визначає вирівнювання вмісту елемента DIV. Атрибут може приймати значення: left, right, center, justify.

Елемент-адреса

<ADDRESS> – текст, що міститься між початковим і кінцевим тегами, оформляється як поштова адреса. Найчастіше оформлення виражається у виділенні рядка адреси курсивом.

! Internet Explorer версії 3.0 неправильно інтерпретує кінцевий тег </ADDRESS>, тому весь текст, що перебуває після адреси, буде відображатися курсивом. Цю помилку можна легко виправити, поставивши після </ADDRESS> будь-який елемент форматування тексту.

Елемент-цитата

<BLOCKQUOTE> – оформляє текст як цитату. Використовується для довгих цитат (на відміну від елемента CITE). Текст, що цитується, відображається окремим абзацом зі збільшеним відступом.

Горизонтальна лінія

<HR> – вставляє в текст горизонтальну розділову лінію.

Атрибути:

WIDTH – визначає довжину лінії в пікселях або відсотках від ширини вікна браузера;

SIZE – визначає товщину лінії в пікселях;

ALIGN – визначає вирівнювання горизонтальної лінії. Атрибут може приймати наступні значення: left, right, center;

NOSHADOW – визначає спосіб зафарбування лінії як суцільний. Атрибут є прапором і не вимагає вказівки значення. Без цього атрибута лінія відображається об'ємною;

COLOR – визначає кольори лінії. Задається або RGB-значенням у шістнадцятиричній системі, або одним з 16 базових кольорів (атрибут працює тільки в Internet Explorer).

Приклад:

І запитала крихітка:

- Що таке

ДОБРЕ

і що таке

ПОГАНО?»

Контрольні запитання

1. В якій структурній частині HTML-документа вводяться текстові дані?
2. За допомогою яких тегів ми можемо відрегулювати відступи від країв сторінки? У чому їхня особливість?
3. Які теги відповідають за форматування абзацу?
4. У чому особливість тегу < PRE >?
5. У чому особливість тегу < ADDRESS >?

Форматування шрифту

Для надання тексту певного начертання – жирний, курсив, підкреслений тощо, розташуйте його між відповідними тегами:

 Жирний текст

<I> *Курсив* </I>

<U> Підкреслений </U>

<STRIKE> ~~Перекреслений~~ </STRIKE>

^{^{Верхній індекс}}

{{Нижній індекс}}

Деякі теги можуть або повинні застосовуватися з певними параметрами, які вказуються в початковому тегу елемента (можна відразу вказувати кілька параметрів в одному тегу). Наприклад, тег (обов'язковий) може мати кілька атрибутів.

Атрибути:

SIZE – задає розмір тексту (за замовчуванням розмір тексту дорівнює 3). Помістивши текст між тегами , де n – цифрове значення від 1 до 6 за зростанням кеглю, можна надати йому потрібний розмір;

FACE – задає стандартне ім'я шрифту. Використовуйте шрифти, які найвірогідніше встановлені на комп'ютері користувача, у протилежному випадку браузер буде використовувати шрифт, установлений за замовчуванням (звичайно Times New Roman). До стандартних шрифтів можна віднести шрифти, що поставляються з Windows 98, Ms Plus, Ms Office. Можна вказувати кілька шрифтів через кому. Якщо у користувача на комп'ютері

немає першого за списком шрифту, браузер буде відображувати текст у наступному за списком форматі.

Приклад:

Задати текст розміром – 4 і наступні шрифти для його відображення: Times New Roman, Arial, Courier, Verdana, Comic Sans MS, Tahoma

 Текст

COLOR – задає колір тексту (за замовчуванням чорний – #000000). Колір тексту може визначатися як самою назвою, наприклад, red, blue тощо, так і бути поданим у шістнадцятиричному виді – #FF0000 (червоний), #0000FF (синій).

Приклад:

Задати колір даного відрізка тексту – червоний:

 Червоний

 Червоний

Виділення тексту кольором

За допомогою елемента STYLE тегу (тег обов'язковий) можна задавати виділення тексту будь-яким кольором:

 Зелений

 Жовтий

 Світло-синій

Завдання для самостійної роботи

1. Підготуйте документ до роботи – наберіть стандартну структуру документа. Продовжуйте роботу всередині елемента BODY.
2. Задати текст розміром – 2, колір – зелений і наступні шрифти для його відображення: Times New Roman, Arial, Courier.
3. Уведіть довільний абзац тексту, що буде виводитися шрифтом, заданим за замовчуванням. Почніть цей абзац із тегу <P>. Виділіть уведений абзац жовтим кольором.
4. Задайте такий текст:

«Правила дій над векторами, заданими своїми координатами:

Нехай у базисі $(i;j)$ задані вектори $a=(x_1;y_1)$ і $b=(x_2;y_2)$:

Координати суми двох векторів: $a+b=(x_1+x_2; y_1+y_2)$

Модуль вектора $a=(x;y)$ обчислюється за формулою:

$$\sqrt{x^2 + y^2}$$

*Знак квадратного кореня вводиться таким кодом – «√»

Контрольні запитання

1. Які теги відповідають за форматування начертання тексту?
2. Яка структура тег + атрибут задає розмір шрифту?
3. Чи можуть теги бути вкладеними? Які, наведіть приклади?
4. Яким способом можна задати колір тексту?
5. Спосіб виділення тексту кольором?

Списки

Пронумеровані списки

У випадку, коли вставлений пронумерований список, браузер автоматично буде проставляти числа перед кожним елементом списку. Починається пронумерований список з тегу `<OL>` і завершується тегом `</OL>`.

Тег `<OL>` може мати наступні параметри:

TYPE – вид лічильника;

A – великі латинські літери;

a – маленькі латинські літери;

I – великі римські цифри;

i – маленькі римські цифри;

1 – арабські цифри;

Атрибут **START** – установлює число, з якого буде починатися відлік.

<code></code>	
<code> Один</code>	1. Один
<code> Два</code>	2. Два
<code> Три</code>	3. Три
<code> Чотири</code>	4. Чотири
<code> П'ять</code>	5. П'ять
<code></code>	

Приклад:

Відобразити список, пронумерований великими латинськими цифрами, який починається з восьми:

<code><OL TYPE="I"></code>	
<code>START="8"></code>	
<code> Вісім</code>	VIII. Вісім
<code> Дев'ять</code>	IX. Дев'ять
<code> Десять</code>	X. Десять
<code></code>	

Марковані списки

Для пронумерованих списків браузер звичайно використовує маркери для позначення елемента списку. Список

починається тегом `<UL>` і завершується тегом `</UL>`. Кожний елемент списку починається з тегу `<LI>`. Тег `<UL>` може включати атрибут `TYPE` зі значеннями `disc`, `circle`, `square`.

<code><UL TYPE=disc> disc </code>	• disc
<code><UL TYPE=circle> circle </code>	o circle
<code><UL TYPE=square> square </code>	§ square

Дані значення – це зовнішній вигляд маркера, що за замовчуванням ставиться у вигляді диска, тобто `disc`.

Приклад:

Відобразити маркований список, тип маркера – за замовчуванням.

<code></code>	
<code> Один</code>	o Один
<code> Два</code>	o Два
<code> Три</code>	o Три
<code> Чотири</code>	o Чотири
<code> П'ять</code>	o П'ять
<code></code>	

Список визначень

Список визначень починається тегом `<DL>` і закінчується тегом `</DL>`. Рядок, в якому вказується те, для чого будуть нижче дані визначення починається тегом `<DT>`. Кожний рядок визначення починається з тегу `<DD>`.

<code><DL></code>	
<code><DT> Термін</code>	Термін
<code><DD> Визначення терміна</code>	Визначення терміна
<code><DD> Визначення терміна</code>	Визначення терміна
<code><DD> Визначення терміна</code>	Визначення терміна
<code><DD> Визначення терміна</code>	Визначення терміна
<code></DL></code>	

З елементом `DL` можна використати атрибут `COMPACT` для установки більш компактного розміщення пунктів списку. `<DL COMPACT>`.

Саме так використовується даний атрибут – ніякі числові значення йому не привласнюються:

Завдання для самостійної роботи

1. Підготуйте документ до роботи в програмі Блокнот. Наберіть основну структуру.

2. Вставте в документ упорядкований (нумерований) список. Нумерація – великі римські, починається із другого номера. Перегляньте результат. Змініть нумерацію на арабські, початок з першої. Перегляньте результат.
3. Вставте в документ упорядкований (маркований) список, тип маркера – квадрат. Змініть маркер на інший тип. Перегляньте результат. Дослідіть, що відображується за замовчуванням.
4. Вставте в документ список визначень слова «інформація»: «Інформація – це:
 - кількість прийнятих, оброблених або переданих повідомлень, біт (обчислювальна техніка й зв'язок);
 - міра організованості (погодженості, зв'язності, упорядкованості) системи (синергетика);
 - зміст сигналу й, отже, зміст відбиття й зміни взагалі».
 Перегляньте отриманий результат.

Контрольні запитання

1. Яким чином можна вказати вид лічильника в нумерованому списку? Продемонструвати на прикладах: цифрову та літерну нумерації.
2. Яким чином можна задати початок відліку?
3. Які види маркованих списків існують?
4. У чому особливість списку визначень?
5. Чи можна створювати список усередині списку? Якщо так, то як?

Таблиці

Використовуючи таблиці, можна створювати такі ефекти, як верстка в кілька колонок, застосування ефектів поєднання картинки й фону, тонкі лінії на всю ширину або висоту сторінки тощо.

Таблиця починається тегом <TABLE> і завершується - </TABLE>. Тег <TABLE> може включати наступні атрибути:

WIDTH="n" – визначає ширину таблиці в пікселях або відсотках, по умовчуванню ширина таблиці визначається змістом комірки;

BORDER="n" – визначає товщину рамки (за умовчанням n = 0 – таблиця відображається без рамки);

BORDERCOLOR="#FFFFFF" – встановлює кольори окантовки, де за умовчанням #FFFFFF (білий) – шістнадцятиричним значенням кольорів;

BGCOLOR = "#FFFFFF" – установлює кольори фону для всієї таблиці;

BACKGROUND = "image.gif" – заповнює фон таблиці зображенням;

CELLSPACING = "n" – визначає відстань між рамками комірок таблиці в пікселях;

CELLPADDING = "n" – визначає відстань у пікселях між рамкою комірки й текстом;

ALIGN – визначає розташування таблиці (left, center, right) у документі (за замовчуванням таблиця притиснута до лівого краю сторінки);

FRAME = "значення" – управляє зовнішньою окантовкою таблиці, може приймати наступні значення:

void – окантовки немає (значення за замовчуванням);

above – межа лише зверху;

below – межа лише знизу;

hsides – межа зверху й знизу;

vsides – тільки межа ліворуч і праворуч;

lhs – тільки ліва межа;

rhs – тільки права межа;

box - зображаються всі чотири межі;

border - також усі чотири боки;

RULES="n" – управляє лініями, що розділяють комірки таблиці. Можливі значення (n):

none – немає ліній (значення за замовчуванням);

groups – лінії будуть тільки між групами рядів;

rows – тільки між рядками;

cols – тільки між колонками;

all – між усіма рядками й колонками.

Таблиця може включати заголовок, що розташовується між тегами <CAPTION></CAPTION>. Він повинен бути безпосередньо після тегу <TITLE>. До заголовка можливе застосування атрибута **ALIGN**, що визначає його положення щодо таблиці:

top – значення по умовчуванню, заголовок над таблицею по центру.

left – заголовок над таблицею ліворуч;

right – заголовок над таблицею праворуч;

bottom – заголовок під таблицею по центру.

Тепер про рядки й комірки таблиці. Рядки таблиці починаються тегом **<TR>** і завершуються тегом **</TR>**, а кожна комірка таблиці починається тегом **<TD>** і завершується тегом **</TD>**. Наступні атрибути можуть застосовуватися для рядків і комірок:

ALIGN – установлює горизонтальне вирівнювання тексту в комірках рядка (left, center, right);

VALIGN – установлює вертикальне вирівнювання тексту в комірках рядка.

top – вирівнювання по верхньому краю, **center** – вирівнювання по центру (це значення приймається за замовчуванням), **bottom** – по нижньому краю;

BGCOLOR="#FFFFFF" – установлює кольори фону рядка або ячейки;

BACKGROUND="image.gif" – заповнює фон рядка або комірки зображенням;

Наступні атрибути можуть застосовуються тільки для комірок:

WIDTH="n" – визначає ширину комірки в пікселях;

HEIGHT="n" – визначає висоту комірки в пікселях;

COLSPAN="n" – розтягування комірки по горизонталі.

Приклад:

<TD COLSPAN="2"> означає, що комірка буде розтягнута на 2 колонки.

ROWSPAN="n" – розтягування комірки по рядку.

Приклад:

<TD ROWSPAN="2"> означає, що комірка буде розтягнута на два рядки таблиці.

NOWRAP – наявність цього атрибута показує, що текст повинен розміщатися в один рядок.

BACKGROUND="image.gif" – заповнює фон комірки зображенням.

Крім цього, будь-яка комірка таблиці може бути визначена не тегами **<TD></TD>**, а тегами **<TH></TH>** – Table Header (заголовок таблиці). Власне кажучи, це звичайна комірка, але текст уведений всередині цих тегів буде виділений напівжирним шрифтом й відцентрований.

Якщо комірка порожня, то навколо неї рамка не рисується. Якщо рамка все ж такі потрібна навколо порожньої комірки, то в неї треба ввести символний об'єкт «** **» (non-breaking space – пробіл, що не розривається). Комірка, як і раніше, буде порожня,

а рамка навколо неї буде. «** **» – обов'язково повинен набиратися малими літерами й закриватися крапкою з комою.

! Теги, що встановлюють форматування шрифту (, <I>, <FONT SIZE="n", FONT COLOR="#FFFFFF"), необхідно повторювати для кожної комірки.

Приклад:

Задайте таблицю такого вигляду:

Комірка 1	Комірка 2
Комірка 3	Комірка 4

Примітка: У даному прикладі створена таблиця з фіксованою шириною (**WIDTH="200"** пікселей), але краще використовувати відсотки, тому що в цьому випадку розмір таблиці буде змінюватися залежно від розміру вікна.

Така таблиця реалізується наступним кодом:

```
<TABLE BORDER="2" WIDTH="200"
BGCOLOR="#00FF00">
<TR>
<TD> Комірка 1</TD> <TD> Комірка 2</TD>
</TR>
<TR>
<TD> Комірка 3</TD> <TD> Комірка 4</TD>
</TR>
</TABLE>
```

Завдання для самостійної роботи

1. Підготуйте шаблон нового документа до роботи з таблицями.
2. Необхідно помістити після тегу **< BODY>** коментар, у якому говориться про те, що наша сторінка буде містити список телефонних номерів.
3. Створіть таблицю з заповненням наступного виду:

Ім'я	Адреса	Телефон
------	--------	---------

...		...
На першому поверсі є безкоштовний телефон		

- Вивчіть, як створена таблиця відображається в програмі Internet Explorer, звертаючи особливу увагу на вплив заданих атрибутів. Змініть ширину вікна оглядача й зверніть увагу, як при цьому змінюється зовнішній вигляд таблиці.
- У новому документі створіть таблицю визначень тегів, що відносяться до форматування шрифту. Але в цьому випадку необхідно створити таблицю з невидимими внутрішніми роздільниками:

<	Жирний текст	
<I>	<i>Курсив</i>	</I>
<U>	<u>Підкреслений</u>	</U>
<STRIKE>	Перекреслений	</STRIKE>
^{	Верхній індекс	}
_{	Нижній індекс	}

*Символ «<>» вводимо кодом «<», «>» – «>».

- Пофарбуйте комірки таблиці: стовпець тегів – зелений, а стовпець визначень – жовтий. Установіть кольори зовнішньої окантовки – коричневий. Перегляньте отриманий результат.

Контрольні запитання

- Для чого використовується табличний формат в HTML-документах?
- Яким чином можна задати заголовок і назву таблиці? Чим відрізняються між собою ці операції?
- Як можна задати порожню комірку?
- Яка структура задання таблиці?

Зображення

У більшості випадків графічні зображення є невід'ємною частиною документа. На сьогодні використовуються два основних формати: *.gif, *.jpg,. Обираючи формат зображення, по-перше беруть до уваги обсяг файла, по-друге – якість зображення, оскільки під час завантаження сторінки лівова частина часу приділяється завантаженню зображення.

Тепер про фонову картинку. Фоном може бути як великий графічний файл, так і його фрагмент, що більш розумно. При використанні останнього, він буде автоматично розмножений браузером, тому підбирайте фрагменти, що добре поєднуються. Вставити в сторінку фонову картинку можна так:

<BODY BACKGROUND="images.gif">

Параметру **BACKGROUND** привласнене значення **images.gif** – це ім'я графічного файла з розширенням. Графічний файл повинен бути розташований в одній папці з Вашим документом, інакше потрібно вказати до нього шлях. Проілюструємо на прикладі файлу **images.gif**.

Приклад:

На два рівні вниз:

Параметр = "Folder_1/Folder_2/images.gif"

На два рівні нагору:

Параметр = ".././images.gif"

!Подібна вказівка шляхів застосовується для різних елементів, наприклад, посилань, малюнків, файлів.

Для вставки зображень у HTML документ використовується наступна конструкція (подана повністю):

```
<IMG SRC="Малюнок" BORDER="0" ALIGN="Вирівнювання"
WIDTH="Ширина" HEIGHT="Висота" HSPACE="Відступ_1"
VSPACE="Відступ_2" ALT="Підказка" NAME="Ім'я"
LOWSRC="Малюнок_2">
```

Атрибути:

SRC – ім'я графічного файла з розширенням (підтримуються *.jpg, *.gif, *.png). Якщо файл перебуває в іншому каталозі, укажіть шлях до нього;

ALIGN – вирівнювання зображення в документі

top або **texttop** - вирівнюють верхній край зображення з верхньою лінією поточного текстового рядка;

middle – вирівнює базову лінію поточного текстового рядка із центром зображення;

absmiddle – вирівнює центр поточного текстового рядка із центром зображення;

bottom або **baseline** - вирівнюють нижній край зображення з базовою лінією поточного текстового рядка;

absbottom – вирівнює нижній край зображення з нижнім краєм поточного текстового рядка;

BORDER – рамка навколо малюнка (значення за замовчуванням - 0 (без рамки));

WIDTH – ширина зображення в пікселях;

HEIGHT – висота зображення в пікселях;

HSPACE – горизонтальний відступ графічного зображення в пікселях (не обов'язковий параметр);

VSPACE – вертикальний відступ у пікселях (необов'язковий параметр);

ALT – це повідомлення, що виводиться замість картинки, якщо вона не показується (елемент не знайдений або користувач настроїв свій браузер так, що той не показує картинку). Крім того, ви побачите цей текст у вигляді підказки, коли курсор миші перебуває на малюнку;

NAME – визначає ім'я зображення (для звичайного зображення, не пов'язаного ні з чим, цей параметр зовсім необов'язковий).

LOWSRC – ім'я графічного файлу з розширенням з альтернативним зображенням більш низької якості (і відповідно меншого обсягу), чим зображення, зазначене в параметрі **SRC**. Оглядачі, що підтримують даний параметр, спочатку завантажують картинку з **LOWSRC**, а потім заміняють її картинкою з **SRC**. (не обов'язковий параметр).

! Завжди задавайте розміри картинки в параметрах HEIGHT й WIDTH, резервуючи тим самим місце у вікні Оглядача ще до завантаження зображення. Параметр ALT можна не вказувати, але раптом у людини не завантажилася картинка, він хоч буде знати, що вона собою представляє.

Графіка може бути статична й динамічна:

Графіка (статична) – це картинка або фото, що мають альтернативний текст (пояснювальний).

Приклад коду:

```
<HTML>
<HEAD>
<TITLE>Заголовок</TITLE>
<BODY>
```

```
<IMG SRC="filename.gif" WIDTH="201"
HEIGHT="120" ALT="ваш текст">
</BODY>
</HTML>
```

Графіка (динамічна) – це анімація або просто мультики, що теж мають альтернативний текст. В Інтернет поширена анімація двох типів "gif" й "flash". Причому "gif"-формат має функцію альтернативного тексту, а "flash" – не має.

Анімаційні елементи можуть бути вставлені на сторінку за допомогою коду подібного до вставки графіки. Але треба враховувати, можливість використання атрибута альтернативного тексту.

У кожному разі **графічний файл** не рекомендується використовувати більше 30 kb через довге завантаження.

Завдання для самостійної роботи

1. Підготуйте шаблон нового документа для роботи з малюнками. Підготуйте малюнок, який ви будете вставляти в документ (не більше 30 кб).
2. Вставте малюнок, вирівнюючи нижній край зображення з базовою лінією поточного текстового рядка. Перегляньте результат. Поміняйте розташування малюнка на сторінці, вирівнюючи верхній край зображення з верхньою лінією поточного текстового рядка. Перегляньте результат.
3. Задайте ширину малюнка – 200 пікс., висоту – 150 пікс., а відступи з боків і ліворуч/праворуч – 20 пікс.
4. Створіть підказку про зміст картинку за допомогою відповідного тегу. Створіть назву до зображення тегом. Перегляньте отриманий результат.

Контрольні запитання

1. Який тип графічних файлів підтримується в HTML документах? Їхні особливості.
2. Яким чином можна встановити картинку фоном сторінки? Які умови повинна задовольняти картинка?
3. Як і для чого створюється коментар (підказка) до картини?
4. Які атрибути тегу вставки зображення є необхідними?
5. Чим відрізняється статична графіка від динамічної?

Гіперпосилання

Створення звичайного гіперпосилання

Посилання HTML дозволяють зв'язати текст або картинку з іншими гіпертекстовими документами. Текст, як правило, виділяється кольорами або оформляється підкресленням. Приклад звичайної конструкції посилання:

```
<A HREF="URL" TARGET="Вікно"  
TITLE="Підказка">Назва посилання</A>
```

Параметри елемента **<A>** такі:

HREF – URL (уніфікований локатор ресурсів) – адреса будь-якого файлу в Інтернет. Може бути абсолютним, тобто вказується повна адреса сторінки (наприклад, <http://lenininc.narod.ru/index.html>) і відносним, як видно з назви – вказується файл щодо поточного (наприклад, index.html).

TARGET – визначає у якому вікні (фреймі) завантажити гіперпосилання. Може мати значення:

top – завантажує гіперпосилання на всьому просторі вікна браузера (якщо до цього існувала розбивка на фрейми, то вона зникне);

blank – завантажує гіперпосилання в новому вікні браузера.

self – завантажує вміст сторінки, у вікно, що містить це посилання (і так використовується за замовчуванням, тому якщо вам треба саме так завантажити посилання, то параметр target взагалі можна не використовувати);

parent – завантажує вміст сторінки, що задана посиланням, у вікно, що є безпосереднім власником набору фреймів;

TITLE – текст підказки, що буде з'являтися при наведенні мишки на гіперпосилання (не обов'язковий параметр).

Приклад:

Створимо посилання на документ 002.htm

```
<A HREF="002.htm" TARGET="self" TITLE="Приклад посилання">  
LINK </A>
```

Після кліку по цьому посиланню відкриється інший документ, у цьому випадку 002.htm.

Графічний файл у ролі посилання

Гіперпосиланням можна зробити й будь-який графічний файл, що підтримується браузером (традиційно – *.gif, *.jpg.). Для цього замість назви посилання потрібно прописати графічний елемент (параметр TITLE повинен бути тільки в малюнка):

Приклад:

```
<A HREF="URL" TARGET="Вікно">  
<IMG BORDER="0" SRC="Малюнок" WIDTH="Ширина"  
HEIGHT="Висота" TITLE="Підказка">  
</A>
```

Посилання на e-mail

Ви можете також створити посилання на e-mail, у цьому випадку потрібно як URL прописати таке:

mailto:адресу електронної пошти

Приклад:

```
<A HREF="mailto:lenin@zeos.net">Моє мило</A>
```

Поштове гіперпосилання має кілька параметрів (необов'язкових):

?subject – тема листа;

&Body – текст вашого повідомлення;

&cc – копії листа через кому;

&bcc – сховані копії листа через кому;

title – "підказка" ставиться за бажанням і розташовується окремо від параметрів поштового посилання (дивіться розташування лапок на прикладі нижче).

Приклад:

```
<a href="mailto:lenin@zeos.net ?subject=Поздоровлення  
&Body=text &cc=copy@mail.ua.net  
&bcc=hidden_copy@mail.ua.net" title="Приклад поштового  
гіперпосилання">Моє мило</a>
```

Після кліку по такому посиланню, відкриється вікно поштової програми користувача з уже заповненими полями. Залишиться тільки написати листа і відправити.

Закладки

Іноді виникає питання: як зробити посилання на певне місце в тому самому або іншому документі, щоб, нажавши на яке-небудь посилання, можна було потрапити в певне місце даного документа. Тут і допоможуть закладки.

Посилання на закладку в тому ж документі мають такий вигляд:

```
<A HREF="#Ім'я закладки">Назва розділу</A>
```

А так виглядає посилання на закладку в іншому документі:

```
<A HREF="Ім'я документа#ім'я закладки">Назва розділу</A>
```

Сама закладка буде такою: ****

Клікнувши на назву розділу, користувач потрапить на вказану Вами закладку.

Завдання для самостійної роботи

1. Підготуйте документ для роботи в програмі Блокнот. Починаємо роботу в елементі <BODY>. Складіть список назв улюблених творів художньої літератури (вірші або проза). Не менш 5 найменувань. Нижче (поза списком) запишіть цитати із цих творів, уривки, коротку характеристику або твір повністю (якщо це вірш). Приведіть до відповідності назви уривків (цитат) з їх же назвами, зазначеними в списку. Створіть гіперпосилання-закладки від елементів списку до відповідних уривків твору. Перегляньте результат
2. Створіть гіперпосилання на інший документ зі слів «інший документ». Документ довільної природи. Перегляньте результат.
3. Створіть елемент-картинку (не більше 30 кб) з підказкою про її вміст і бажано з поштовою тематикою. Створіть гіперпосилання з картини на e-mail (на свій або на e-mail свого знайомого). Перегляньте результат.
4. Налаштуйте кольори посилання до, під час і після користування нею. Перегляньте результат.

Контрольні запитання

1. Для чого потрібні гіперпосилання?
2. Чим відрізняється посилання від закладки?
3. Який спосіб оформлення закладки?
4. Які варіанти завантаження гіперпосилання Ви знаєте?
5. Чи можна використати в гіперпосиланні підказку (роз'яснювальний текст)? Якщо так, то як?
6. Чи можна як посилання використовувати графічний файл? Якщо так, то як це зробити?

Стилі

CSS Cascading Style Sheets (Таблиці каскадних стилів) – це набір правил оформлення й форматування, що може бути застосований до різних елементів сторінки.

У стандартній мові HTML для присвоєння якому-небудь елементу певних властивостей (таких, як кольори, розмір,

положення на сторінці тощо.) доводиться щораз описувати ці властивості. Застосовуючи CSS, Ви можете один раз описати властивості елементів і визначити цей опис як стиль, а надалі просто вказувати, що елемент, який ви хочете оформити відповідно, повинен прийняти властивості стилю, описаного вами.

Ви можете зберегти опис стилю не в тексті вашої сторінки, а в окремому файлі – це дозволить використовувати опис стилю на будь-якій кількості Web-сторінок.

Описи стилів перебувають у тегах <STYLE></STYLE> і розміщуються між тегами <HEAD></HEAD>. У таблиці наведені деякі властивості елементів, керованих за допомогою CSS.

Властивості шрифту:	
font-family	Використовується для вказівки шрифту або шрифтового сімейства, яким буде відображатися елемент. Приклад: P {font-family: Verdana, sans-serif;}
font-weight	Визначає ступінь жирності шрифту за допомогою трьох параметрів: lighter (звичайний), bold (жирний), bolder (дуже жирний) Приклад: B {font-weight: bolder;}
font-size	Установлює розмір шрифту. Параметр може вказуватися у відсотках, пікселях або сантиметрах. Приклади використання для тегів H1 , H2 , H3 : Приклад: H1 {font-size: 200%;} H2 {font-size: 150px;} H3 {font-size: 400pt;}
font-variant	Визначає, чи буде шрифт виведений у вигляді малих прописних букв. Значення: • normal – звичайне накреслення (за замовчуванням). • small-caps – виводить шрифт у вигляді малих

	прописних букв. inherit – застосовується значення батьківського елемента. Приклад: P {font-variant: small-caps }
font-stretch	Визначає товщину шрифту. Значення: normal – звичайна ширина (за замовчуванням) wider – збільшення поточної ширини. narrower – зменшує поточну ширину на одиницю. ultra-condensed – найменше значення ширини. extra-condensed – значення більше, ніж попереднє condensed – значення більше, ніж попереднє. semi-condensed – значення більше, ніж попереднє. semi-expanded – значення більше, ніж при звичайній товщині. expanded – значення більше, ніж попереднє. extra-expanded – значення більше, ніж попереднє. ultra-expanded – максимальне значення ширини. inherit – застосовується значення батьківського елемента. Приклад: BODY{ font-stretch: condensed }
font	Зручна властивість для установки відразу всіх параметрів шрифту. Якщо якісь значення пропущені, то приймається їхнє значення за замовчуванням. Значення: font-style – тип шрифту; font-variant – значення, що визначає вивід шрифту у вигляді малих прописних букв;

	font-weight – товщина; font-size – кегль; line-height – інтерліньяж; font-family – сімейство шрифтів; inherit – застосовується значення батьківського елемента. Приклад: P {font: oblique 12pt "Helvetica Nue", serif; font-stretch: condensed }
Властивості тексту:	
text-decoration	Установлює ефекти оформлення шрифту, такі, як підкреслення або закреслювання тексту. Приклад використання для тегу H4: Приклад: H4 {text-decoration: underline;} (підкреслення) H4 {text-decoration: line-through;} (закреслювання)
text-align	Визначає вирівнювання елемента. Приклад: P {text-align: left} (вирівнювання по лівому краю) P {text-align: right} (вирівнювання по правому краю) P {text-align: justify} (вирівнювання по ширині) P {text-align: center} (вирівнювання по центру)
text-indent	Установлює відступ першого рядка тексту. Найчастіше використовується для створення параграфів з табульованим першим рядком. Приклад використання для тегу H1: Приклад: H1 {text-indent: 60pt;}
line-height	Управляє інтервалами між рядками тексту. Приклад: P {line-height: 50 %}

text-transform	Автоматично переводить у тексті букви у верхній (нижній) регістр. Значення: • none – відсутність зміни регістру (за замовчуванням); • capitalize – переводить першу букву кожного слова у верхній регістр; • uppercase – переводить всі букви у верхній регістр; • lowercase – переводить всі букви в нижній регістр; • inherit – застосовується значення батьківського елемента. Приклад: P {text-transform: uppercase }
text-shadow	Визначає значення, що встановлюють ефект затінення тексту. Може містити кілька значень, які варто вказати один за одним, розділивши комами. У кожному окремому значенні тінювих ефектів обов'язково повинно бути вказано значення зсуву й можливі такі значення, як радіус різкості й кольори. Значення: • none – відсутність значення (за замовчуванням); • color – колір затінення тексту; • перше числове значення відповідає відстані по горизонталі вправо від тексту. Негативне значення відповідає відстані вліво від тексту; • друге числове значення, відповідає стандарту - відстані по вертикалі вниз від тексту; Негативне значення відповідає відстані вгору від тексту; • третє числове значення, відповідає стандарту – радіусу нерізкості тіні тексту. • inherit – застосовується значення батьківського елемента. Приклад:

	H1{text-shadow: blue 5px 5px 3px}
Зміст і списки За допомогою цих властивостей виконується керування як стилем елементів списків, так й установкою нумерації й маркування.	
list-style-type	Визначає стиль маркерів або номерів списку. Значення: • disc – маркер у вигляді диска (за замовчуванням). • circle – маркер у вигляді окружності. • square – маркер у вигляді квадрата. • decimal – списки, нумеровані за допомогою десятикових чисел, починаючи з 1, • decimal-leading-zero – списки, нумеровані за допомогою десятикових чисел із провідними нулями, наприклад 01, 02, 03 тощо; • lower-roman – списки, нумеровані за допомогою малих римських цифр; • upper-roman – списки, нумеровані за допомогою великих римських цифр; • hebrew – списки, нумеровані за допомогою цифр іврити; • georgian – списки, нумеровані за допомогою грузинських цифр; • armenian – списки, нумеровані за допомогою вірменських цифр; • cjk-ideographic – списки, нумеровані за допомогою ідеограм; • lower-greek – використання малих грецьких букв; • hiragana – використання букв японської абетки хірагана; • hiragana-iroha – використання букв японської абетки хірагана ірода; • katakana-iroha – використання букв японської абетки катакана ірода; • none – немає ні маркерів ні номерів; • inherit – приймається значення батьківського елемента;

	Приклад: ol {list-style: upper-alpha}
Кольори елемента й фону:	
color	Визначає кольори елемента I {color: yellow;} Приклад використання для тегу H3: Приклад: H3 {color: #0000FF;}
background-color	Установлює кольори фону для елемента. Приклад використання для тегу H3: Приклад: <H3 style="background-color:gold; color:brown;"> Приклад </H3>
background-image	Визначає фонове зображення елемента. Значення: none – фонове зображення не встановлюється; - будь-яке значення URL фонового зображення, що відповідає стандарту; inherit – застосовується значення батьківського елемента. Приклад: H1{background-image: url(pictures.gif)}
background-attachment	Визначає, чи буде фонове зображення зафіксоване у вікні браузера, чи буде прокручуватись разом з документом. Значення: scroll – фонове зображення прокручується разом з документом (за замовчуванням); fixed – фонове зображення фіксується у вікні браузера; inherit – застосовується значення батьківського елемента. Приклад: BODY{background-image: url(pictures.gif); background-attachment: fixed;}
Фільтри	

Важливо відзначити, що фільтри можуть бути додані тільки до елементів HTML, які розглядаються як керуючі. Це будь-які елементи, що створюють прямокутне місце на WEB-сторінці. От деякі з керуючих елементів: BODY, DIV, MARQUEE, TD, IMG, SPAN, TR.	
filter alpha	Фільтр alpha установлює рівень прозорості (працює не з усіма браузерами) filter: alpha(opacity, finishopacity, style,start, start, finish, finish) Значення: opacity – початковий рівень прозорості; finishopacity – кінцевий рівень прозорості; style – стиль градієнта(0 – рівномірна прозорість,1 – лінійний, 2 – радіальний, 3 – прямокутний ; start – початкові координати градієнта по горизонталі/вертикалі; finish – кінцеві координати градієнта по горизонталі/вертикалі; Приклад: DIV{ filter: alpha(opacity=10, finishopacity=90, style=0,start=10, start=10, finish=100, finish=100)}
filter blur	Фільтр blur змазує зображення/текст у певну сторону (працює не з усіма браузерами) filter: blur(Add, direction, strength) Значення: add – об'єкт (0 – результат роботи фільтра, 1 – фільтр буде накладений на вихідний зразок); direction – напрямок змазування об'єкта в градусах (0 – вгору й далі по годинниковій стрілці із кроком 45 градусів); strength – ступінь розмиття; Приклад: IMG{ filter: blur(Add=1, Direction=1, Strength=15)} <IMG STYLE="filter: blur(Add=1, Direction=1,

	Strength=15)" SRC="*.jpg" WIDTH="198" HEIGHT="140"> <P STYLE="filter:Blur(Add='1', Direction='30', Strength='10'); WIDTH: 200; HEIGHT: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Blue;"> Приклад </P>
filter chroma	Фільтр chroma робить задані кольори прозорими (працює не з усіма браузерами) filter: chroma(color) Значення: color – колір, що буде прозорим. Приклад: IMG{ filter: chroma(color=#6c75d2)}
filter dropshadow	Фільтр dropshadow створює ефект підвішеності (працює не з усіма браузерами) filter: dropshadow(color, offx, offy, positive) Значення: color – колір тіні; offx - зсув тіні щодо тексту по осі X; offy – зсув тіні щодо тексту по осі Y; positive – характер освітлення (0 або 1). Приклад: DIV{filter: dropshadow(color=silver, offx=3, offy=3,positive=0) } <P STYLE="filter:DropShadow(Color=' #7FFF00',Off='3',Off='3',Positive='1'); width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Green;"> Приклад </P>
filter shadow	Створює тінь тексту Синтаксис: STYLE="filter:Shadow(Color='Color', Direction='Direction')" Color – колір тіні; Direction – напрямок тіні (від 0 до 315).

	Приклад: <P STYLE="filter:Shadow(Color=' #000000', Direction='130'); width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Red;"> Приклад </P>
filter flip	Фільтр fliph перевертає об'єкт горизонтально. Приклад: IMG{ filter: fliph } <P STYLE="filter:Flip; width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Red;"> Приклад </P>
filter flipv	Фільтр flipv перевертає об'єкт вертикально. (працює не з усіма браузерами) Приклад: IMG{ filter: flipv }
 <P STYLE="filter:Flip; width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Red;"> Приклад </P>
filter glow	Фільтр glow додає світіння уздовж зовнішніх границь об'єкта, створюючи ефект "загоряння" границь об'єкта (працює не з усіма браузерами) filter: glow(strength, color) Значення: stregth – сила світіння в діапазоні 0-100; color – колір світіння. Приклад: <P STYLE="filter:Glow(Strength='3', Color=' #00FF00'); width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: DarkGreen;"> Приклад </P>
filter mask	Фільтр mask відображає текст так, начебто він виділений мишею. (працює не з усіма браузерами) filter:mask(Color)

	Значення: Color – колір, що оточує текст. Приклад: <code>IMG{ filter:mask(color=red) }</code> <code><P STYLE="filter:Mask(Color='Red'); width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center;"> Приклад </P></code>
filter wave	Робить текст хвилястим (працює не з усіма браузерами) Синтаксис: <code>STYLE="filter: Wave(Freq='Freq', Add='Add', LightStrength='LightStrength', Phase='Phase', Strength='Strength')"</code> Freq - кількість хвиль Add - показати/сховати окантовку (1 або 0 відповідно) LightStrength - сила хвиль Phase - кут хвиль Strength - інтенсивність хвиль Приклад: <code><P STYLE="filter:Wave(Freq='5', Add='0', LightStrength='1', Phase='2', Strength='2'); width: 200; height: 28; font-size: 16pt; font-weight: bold; text-align: center; color: Green;"> Приклад </P></code>
Властивості границь:	
margin-left (ліворуч) margin-right (праворуч) margin-top (зверху) margin-bottom (знизу)	Установлюють значення відступів навколо елемента. Приклад використання для малюнка: <code>IMG { margin-left: 20pt}</code> <code>IMG { margin-right: 20pt}</code> <code>IMG { margin-top: 20pt}</code> <code>IMG { margin-bottom: 20pt}</code>
Одиниці виміру:	
Px	Пікселі
Cm	Сантиметри

mm	Міліметри
pt або %	Відсотки

Приклад 1

Використання стилю для конкретного слова – "Приклад". Воно повинне відображатися шрифтом Comic Sans MS (якщо такого немає на машині користувача, використовується шрифт Tahoma), жирним, розміром 25 відсотків і темно-зеленим кольором:

```
<!--Описуємо стилі під ім'ям Example-->
<STYLE><!--
.Example{font-family: Comic Sans MS, Tahoma; font-weight:
bolder; font-size: 25pt; color: darkgreen;}
-->
</STYLE>
<!--Тут викликаємо опис стилів-->
<div class="Example"> Приклад </div>
```

Якщо Ви розташовуєте стилі в окремому файлі, тоді між тегами <HEAD></HEAD> кожного HTML документа потрібно додати посилання на CSS файл:

```
<LINK REL=STYLESHEET TYPE="text/css" HREF="*.css">

*.css – це Ваш CSS файл, що містить опис стилів, які застосовані. Якщо він перебуває в іншому каталозі, потрібно вказати до нього шлях. Створюється CSS файл у будь-якому текстовому редакторі, наприклад, у Блокноті, потрібно лише змінити розширення текстового файлу на CSS. У CSS файлі не повинні вказуватися теги <STYLE></STYLE>.
```

Приклад 2

У нас є файл Example.css, що містить такі параметри (підкреслюємо посилання тільки при наведенні на них курсором):

```
<!--
A { text-decoration: none; }
A:hover { color: #FF0000; text-decoration: underline; }
-->
```

Посилання на нього в HTML документі буде виглядати так:
<LINK REL=STYLESHEET TYPE="text/css" HREF="Example.css">

Приклад 3

Створити таблицю стилів наступного виду: посилання – без підкреслення; абзаци – блакитного кольору, 12-го розміру й гарнітура – Arial; заголовки – червоного кольору, 18-го розміру.

```
<HEAD>
<TITLE> Приклад використання таблиці стилів </TITLE>
<!-- Вставляємо таблицю стилів -->
<STYLE TYPE="text/css" TITLE="Cool table">
<!--
A {text-decoration : none;}
P {color : blue; font-size : 12pt; font-family : Arial;}
H1 {color : red; font-size : 18pt;}
-->
</STYLE>
<!-- ... закінчили вставляти -->
</HEAD>
```

Ви можете визначити стиль для будь-якого тегу окремо. Для цього потрібно в тег додати елемент STYLE й описати його стиль у лапках. Наступний приклад відображає слово "Приклад" шрифтом Verdana, розміром 150 відсотків і червоним кольором.

```
<H3 STYLE="font-family:Verdana; font-size:150%; color:red">
Приклад </H3>
```

Завдання для самостійної роботи

1. Створіть наступну таблицю стилів для вашого документа: посилання – без підкреслення, підкреслення тільки під час звернення; абзаци – коричневого кольору, 14-го розміру I гарнітура – Arial; заголовки – червоного кольору, шрифт – жирний 20-го розміру. Перегляньте результат, що вийшов.
2. Створіть таблицю стилів для конкретно виділеного в тексті слова: воно повинне відображатися гарнітурою Comic Sans MS (якщо такої немає на машині користувача, використовується шрифтTahoma), жирним шрифтом, розміром 28 і жовтогарячим кольором. Перегляньте результат.
3. Створіть таблицю стилів в окремому файлі: посилання – без підкреслення, підкреслення тільки під час звернення; абзаци – червоного кольору, жирні, 12-го розміру та гарнітура – Times New Roman; заголовки – червоного кольору, шрифт – дуже жирний з підкресленням, 20-го розміру, вирівнювання – по центру. Переглядаємо результат.
4. Створіть коментарі до таблиці стилів з повним поясненням.

Контрольні запитання

1. Що таке CSS Cascading Style Sheets (Таблиці каскадних стилів)? Для чого їх використовують?
2. Де і яким чином можна розмістити CSS?
3. У чому особливість створення CSS в окремому документі?
4. Що можна настроїти за допомогою CSS?

Селектори

Це елементи дескрипторів, що задають на початку визначення таблиць стилів; ці елементи повідомляють браузеру, де застосувати стиль. Те, що стоїть після селектора визначення стилю, поміщається у фігурні дужки. От приклад, у якому селектором є BODY:

```
body { color: blue}
```

Кілька селекторів можна групувати разом, відокремивши їх один від одного комами:

```
h1, p, h2 {font-size: 12px}
```

Замість селекторів можна використовувати маску «*». У даному прикладі стиль із певним кеглем шрифту застосовується до всього, що є дескриптором на сторінці:

```
* { font-size: 14pt}
```

Іншим символом маски є знак «>». Таким чином, браузеру дається вказівка шукати дочірні селектори в межах певного батьківського. У даному прикладі стиль застосовується тільки до елементів LI списку OL:

```
ol > li {list-style-type: decimal}
```

Використовуючи селектори класів, можна до одного й того самого дескриптора застосовувати різні стилі. Після спільного селектора ставиться крапка та ім'я класу. Стиль застосовується до того дескриптора, атрибут CLASS якого відповідає цьому імені.

У наступному прикладі стиль застосовується до всіх дескрипторів H2, у яких атрибут CLASS дорівнює "myblue":

```
h2.myblue {background-color: blue}
```

```
<h2 class="myblue">у цього заголовка синє тло</h2>
```

Псевдокласи

Псевдокласи є особливою групою, що дозволяє поєднувати кілька стилів для якого-небудь об'єкта. Наприклад, ви можете задати властивості для першої букви параграфа. Для цього ви призначаєте для дескриптора **P** псевдоклас **:first-letter**, у якому встановлюєте різні стилі:

p:first-letter { float: right; font-size: 20%; color: red;}

У CSS2 визначаються такі псевдокласи:

:first-child – перший дочірній елемент іншого елемента;
:link – посилання, до яких ще не зверталися;
:visited – посилання, до яких вже зверталися;
:hover – елемент, над яким у цей час перебуває курсор;
:active – активний у цей момент елемент ;
:lang – цей псевдоклас визначає поточна мова;
:first-line – перший рядок абзацу;
:first-letter – перша буква абзацу;
:before – визначає вміст перед елементом;
:after – визначає вміст після елемента.

Форми

Форма – це інструмент, за допомогою якого HTML-документ може відправити інформацію у визначене місце в мережі. Форми застосовують для опитування відвідувачів, покупки чого-небудь, відправлення електронної пошти тощо.

Принцип роботи форм такий: відвідувач, що зайшов до вас на сторінку, заповнює форму, а після натискання певної кнопки форма бере дані з заповнених полів і відправляє їх у призначене місце.

Форми розміщаються між тегами **<FORM></FORM>**. HTML-документ може містити в собі кілька форм, але вони не повинні перебувати одна всередині іншої. Тег **<FORM>** може містити такі атрибути:

ACTION – обов'язковий атрибут. Визначає, де перебуває оброблювач форми.

METHOD – визначає, яким образом дані з форми будуть передані оброблювачеві. Припустимі значення: **METHOD = post** й **METHOD = get**. Якщо значення атрибута не встановлено, за умовчанням передбачається **METHOD=GET**.

ENCTYPE – визначає, яким чином дані з форми будуть закодовані для передачі оброблювачеві.

Для внесення інформації користувачем у форму використовується елемент **<INPUT>**. Це і є поля, у які користувач вводить інформацію. Кожен елемент **<INPUT>** включає атрибут **NAME = ім'я**, що визначає ім'я даного поля (ідентифікатор поля). У таблиці представлені основні типи елементів **<INPUT>**:

TYPE=text	Визначає вікно для введення рядка тексту. Може містити додаткові атрибути SIZE=число (ширина вікна введення в символах) і MAXLENGTH=число (максимально допустима довжина рядка, у символах): <INPUT TYPE=text SIZE=20 NAME=User VALUE="LENIN INC"> LENIN INC Визначає вікно шириною 20 символів для введення тексту. За замовчуванням у вікні перебуває текст LENIN INC, який користувач може змінити.
TYPE=password	Визначає вікно для введення пароля. Абсолютно аналогічний типу text, тільки замість символів тексту, що вводиться, відображає на екрані зірочки (*): <INPUT TYPE=password NAME=PW SIZE=20 MAXLENGTH=10> Визначає вікно шириною 20 символів для введення пароля. Максимально припустима довжина пароля - 10 символів.
TYPE=radio	Визначає радіокнопку. Може містити додатковий атрибут CHECKED (показує, що кнопка помічена). У групі радіокнопок з однаковими іменами може бути тільки одна позначена: <INPUT TYPE=radio NAME=Question VALUE="Yes" CHECKED> Так <INPUT TYPE=radio NAME=Question VALUE="No"> Немає

	<INPUT TYPE=radio NAME=Question VALUE="Possible"> Можливо ☐ Так ☐ Немає ☐ Можливо Визначає групу із трьох радіокнопок, підписаних Yes, No й Possible. Спочатку позначена перша із кнопок. Якщо користувач не позначить іншу кнопку, оброблювачу буде передана змінна Question зі значенням Yes. Якщо користувач відзначить іншу кнопку, оброблювачу буде передана змінна Question зі значенням No або Possible.
TYPE=checkbox	Визначає квадрат (прапорець-індикатор), у якому можна зробити позначку. Може містити додатковий атрибут CHECKED (показує, що квадрат позначений). На відміну від радіокнопок, у групі квадратів з однаковими іменами може бути кілька позначених квадратів: <INPUT TYPE=checkbox NAME=Comp VALUE="CPU"> Процесори <INPUT TYPE=checkbox NAME=Comp VALUE="Video" CHECKED> Відеоадаптери <INPUT TYPE=checkbox NAME=Comp VALUE="Scan"> Сканери <INPUT TYPE=checkbox NAME=Comp VALUE="Modem" CHECKED> Модеми ☐ Процесори ☒ Відеоадаптери ☐ Сканери ☒ Модеми Визначає групу із чотирьох квадратів. Спочатку позначені другий і четвертий

	квадрати. Якщо користувач не зробить змін, оброблювачу будуть передані дві змінні: Comp=Video й Comp=Modem.
TYPE=hidden	Визначає схований елемент даних, що не видний користувачу під час заповнення форми і передається оброблювачу без змін. Такий елемент іноді корисно мати у формі, що час від часу піддається переробці, щоб оброблювач міг знати, з якою версією форми він має справу. <INPUT TYPE=hidden NAME=version VALUE="1.1"> Визначає сховану змінну version, що передається оброблювачеві зі значенням 1.1.
TYPE=submit	Визначає кнопку, при натисканні на яку запускається процес передачі даних з форми оброблювачу: <INPUT TYPE=submit VALUE="Відправити"> Отправить
TYPE=reset	Визначає кнопку, під час натискання на яку очищуються поля форми. Оскільки при використанні цієї кнопки дані оброблювачу не передаються, кнопка типу reset може й не мати атрибута name: <INPUT TYPE=reset VALUE=" Скидання "> Оброс

Форми можуть містити поля для введення великого тексту

<TEXTAREA>:

<TEXTAREA NAME=address ROWS=5 COLS=50>

Набери тут повідомлення

</TEXTAREA>



Атрибут **NAME** визначає ім'я, під яким уміст вікна буде передано оброблювачу. Атрибут **ROWS** установлює висоту вікна в рядках. Атрибут **COLS** установлює ширину вікна в символах. Текст, розміщений між тегами **<TEXTAREA>...</TEXTAREA>**, є вмістом вікна за замовчанням. Користувач може його відредагувати або просто стерти.

Форми можуть містити меню вибору, що починається тегом **<SELECT>** (містить обов'язковий атрибут **NAME**, що визначає ім'я меню) і завершується **</SELECT>**. Між ними перебувають теги **<OPTION>**, що визначають елемент меню. Обов'язковий атрибут **VALUE** встановлює значення, що буде передано оброблювачу, якщо обрано цей елемент меню. Тег **<OPTION>** може включати атрибут **SELECTED**, який показує, що цей елемент обраний/позначений за замовчуванням.

<pre><SELECT NAME="ім'я"> <OPTION VALUE="option_1" selected>текст 1 <OPTION VALUE="option_2">текст 2 <OPTION VALUE="option_n">текст n </SELECT></pre>	
---	--

Тег **<SELECT>** може також містити атрибут **MULTIPLE**, присутність якого показує, що з меню можна вибрати кілька елементів. Більшість Оглядачів показують меню **<SELECT MULTIPLE>** у вигляді вікна, у якому перебувають елементи меню. Висоту вікна в рядках можна задати атрибутом **SIZE=число**.

<pre><SELECT MULTIPLE SIZE=3 NAME="ім'я"> <OPTION VALUE="option_1" selected>текст 1 <OPTION VALUE="option_2">текст 2 <OPTION VALUE="option_n">текст n </SELECT></pre>	
---	--

Завдання для самостійної роботи

1. Створити новий документ у Блокноті. Створити e-mail форму наступного виду:

Імя:	
Тема:	
Сообщение:	
Отправить Сброс	

Для введення форми використати таблицю, у яку вставити елементи форми.

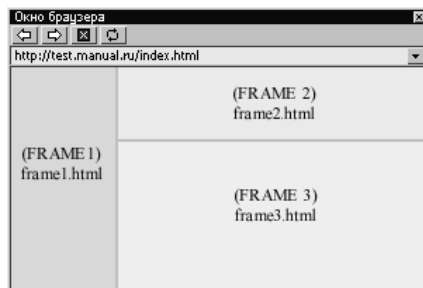
2. Відобразити на сторінці:
 - а) перемикач із 4-х пунктів (Так, Ні, Не знаю, Може бути);
 - б) прапорець – індикатор включених пристроїв з 3 пунктів (принтер, сканер, модем);
 - в) список, що відкривається, з 5 елементів (шрифти);
 - г) розкритий список з 3 елементів (настроювання накреслення: жирний, курсив, підкреслення);
 - д) поле для введення символів (пароль, текст – не більше 5 рядків)

Контрольні запитання

1. Що таке форми і для чого вони використовуються?
2. Як задати форму «Уведення пароля»?
3. Як задати форму «Прапорці» й «Перемикач»? Чим вони відрізняються між собою?
4. Які атрибути обов'язково потрібно вказувати при формуванні елемента для введення тексту?

Фрейми

Фрейми (frames) використовуються для розбивки вікна браузера на кілька зон, кожна з яких – це окремий HTML-документ (фрейм). Як правило, фрейми використовуються для полегшення навігації по сайту, створення навігаційного меню. Проте більшість розроблювачів уникають використання фреймів, маючи, повірте, досить вагомі підстави. Намагайтеся використовувати фрейми тільки тоді, коли це дійсно необхідно.



Елементи для створення фреймів і робота з ними:

FRAMESET	Визначає фреймову (віконну) структуру документа: розміри й розташування фреймів на сторінці
FRAME	Визначає фрейм і його властивості всередині FRAMESET-структури
NOFRAMES	Визначає, що показувати, якщо браузер не підтримує фрейми

Тіло документа замінюється описом фреймів **<FRAMESET>**. Елемент **<BODY>** у такому документі відсутній, а при наявності ігнорується браузером.

Відкриваючий елемент **<FRAMESET>** повинен обов'язково містити атрибут **COLS** або **ROWS**, що визначає спосіб розбивки вікна. У першому випадку вікно розбиває вертикальними лініями, а в другому – горизонтальними. Значення кожного із цих атрибутів – це перерахування розмірів окремих фреймів у відсотках від ширини вікна або в пікселях, остання область може бути задана за допомогою символу «*», який означає, що їй виділяється все місце, що залишилося:

<FRAMESET COLS=«60%,40%»>
<FRAMESET ROWS=«40%,40%,*»>

Між тегами **<FRAMESET>...</FRAMESET>** повинно розташовуватися рівно стільки елементів, скільки областей задано в атрибутах **COLS** й **ROWS**. При цьому можуть використовуватися додаткові елементи **<FRAMESET>**, які описують подальшу розбивку на зони меншого розміру, або непарні теги **<FRAME>**, що визначають спосіб використання області.

<FRAME> повинен містити обов'язковий атрибут **SRC=«ім'я документа»**, за допомогою якого вказується, який документ за-

вантажується у відповідний фрейм. Значення цього атрибута – абсолютна або відносна адреса потрібного документа. Так само виділяють атрибут **NAME=«ім'я»**, що дозволяє задати ім'я створеної області у вигляді послідовності латинських букв і цифр:

<FRAMESRC=«*.htm» NAME= «first»>

Це ім'я можна використовувати, щоб завантажувати нові документи в раніше створену область. Для цього в тег **<A>**, що визначає гіперпосилання, необхідно додати атрибут **TARGET=**, значення якого співпадає з раніше визначеним ім'ям області. Під час переходу за цим гіперпосиланням новий документ завантажиться в зазначений фрейм.

Завдання для самостійної роботи

1. Запустіть текстовий редактор. Уведіть вказаний код:

```
<HTML>
<HEAD>
<TITLE>Опис фреймів< /TITLE > </HEAD>
<FRAMESET ROWS="60%,*">
<FRAME SRC="1.htm">
<FRAMESET COLS="35%, 65%" >
<FRAME SRC="*.htm">
<FRAME SRC="**.htm">
</FRAMESET>
</HTML>
```

Далі виконати такі зміни:

- а) додайте в першому горизонтальному фреймі вертикальну розбивку й задайте відображення в новому фреймі вашого документу; перегляньте;
- б) поміняйте процентне співвідношення між двома нижніми фреймами; перегляньте;
- в) поміняйте вертикальну й горизонтальну розбивку місцями; перегляньте результат;
- г) допишіть для кожного фрейма атрибут – ім'я.

Frame 1	Frame 4		Frame 5
Frame 2			

Frame 3		Frame 6
---------	--	---------

2. Створіть фреймову структуру такого виду:

У кожному з фреймів повинен відображатися який-небудь документ.

Контрольні запитання

1. Яким чином взаємодіють елементи **<BODY>** й **<FRAMESET>**?
2. За допомогою яких атрибутів тегу **<FRAMESET>** можна поділити сторінку на заздалегідь визначені за розміром фрейми?
3. Який елемент дозволяє описати сам фрейм?
4. Чи можуть фрейми бути вкладеними? Чи можна змінити положення границь фреймів методом перетаскування за допомогою миші?
5. Як організовується гіперпосилання у фреймовій структурі?

Навігаційні карти

Карти – це спосіб зробити різні частини одного графічного зображення гіперпосиланнями. Вони дозволяють виділити окремі області зображень і визначити для кожної з них свою дію.

Карти, які оброблюються на клієнтській машині браузером, створюються за допомогою елементів **MAP** й **AREA** і є гіперпосилання, оформлені у вигляді активних зон на зображенні.

Застосувати створену карту до зображення можна викликавши її за ім'ям, користуючись атрибутом **USEMAP** елемента **IMG**.

Приклад:

```
<!-- Створюємо карту з ім'ям ImageMap: -->
<MAP NAME="ImageMap">
  <AREA HREF="something.html" SHAPE="rect"
    COORDS="0,0,70,140" ALT="Ліва половинка">
  <AREA HREF="anything.html" SHAPE="rect"
    COORDS="71,0,140,140" ALT="Права половинка">
</MAP>
<!-- Створили. -->
```

```
<BODY>
<!-- Пригвинчуємо її до зображення -->
<IMG src="/img/block.gif" USEMAP="#ImageMap" HEIGHT="140"
WIDTH="140" BORDER="0">
...
</BODY>
```

У даному прикладі ми створили квадратне зображення розміром 140x140 пікс., ліва частина якого є посиланням на файл *something.html*, а права – на файл *anything.html*.

У HTML-коді цього прикладу виключені посилання на script, що використовувався в прикладі для виводу повідомлень:

```
<IMG SRC="Map_example.jpg" BORDER="0" WIDTH="200"
HEIGHT="160" ALT="Приклад зображення-карти"
USEMAP="#Map_example">
<MAP NAME="Map_example">
  <AREA SHAPE="rect" COORDS="11,11,70,24"
    TITLE="Посилання 1" HREF="URL">
  <AREA SHAPE="rect" COORDS="70,72,128,83"
    TITLE="Посилання 2" HREF="URL">
  <AREA SHAPE="rect" COORDS="190,136,128,149"
    TITLE="Посилання 3" HREF="URL">
```

Як бачите, щоб створити карту, потрібно вставити в тег **** атрибут **USEMAP="#name"**, де **name** – ім'я карти (значок # обов'язковий). У прикладі використовувалася назва **Map_example**. Інші атрибути в цьому тегу випущені, оскільки про них уже розповідалося.

Далі описуємо активні області карти. Починаємо з тега, що відкривається, **<MAP NAME="name">** (тут повторюється ім'я, але вже без значка #), а закінчуємо тегом **</MAP>**. Між цими тегами поміщаємо опис кожної активної області зображення: **<AREA SHAPE="форма" COORDS="координати" HREF="адреса" TITLE="альтернативний текст">**. Елемент **<AREA>** має наступні атрибути:

SHAPE – описує форму виділеної області, можливі значення:

rect – прямокутник;
circle – коло;
poly – багатокутник;

default – визначає всю область, тобто весь малюнок може стати посиланням.

NOHREF – визначає, що даної області не відповідає ніяке посилання. Де це можна використати? Наприклад, якщо ви хочете зробити посилання не у вигляді кола, а у вигляді кільця;

ALT – альтернативний текст для виділеної області, використовується невізуальними браузером (такими, що не відображують картинку);

COORDS – координати, що визначають розміри та положення області на зображенні. Всі координати відраховують у пікселях від лівого верхнього кута зображення. Кількість і порядок значень залежить від значення атрибута SHAPE:

rect – лівий-X, верхній-Y, правий-X, нижній-Y (тобто спочатку координати лівого верхнього кута, потім правого нижнього);
circle – центр-X, центр-Y, радіус (тобто горизонтальна й вертикальна координати центра кола й радіус)
poly – X1, Y1, X2, Y2, ..., Xn, Yn (просто перераховуються координати всіх вершин багатокутника).

TITLE – назва виділеної області, виводиться у вигляді підказки, що спливає при наведенні курсору на область малюнка.

TARGET – визначає вікно (фрейм), на яке вказує гіпертекстове посилання. Цей атрибут використовується тільки разом із атрибутом HREF. Як значення необхідно задати ім'я одного з існуючих фреймів (див. елемент FRAME) або одне із зарезервованих імен, докладно описаних в атрибуті TARGET елемента <A>. Значення цього атрибута ("_top", "_blank", "_self" або "_parent") визначає, у якому вікні буде відкритий документ.

HREF – визначає область як гіпертекстове посилання. Значення задається у вигляді посилання (URL).

Приклад:

<!--Створюємо карту із круглою діркою посередині -->

<MAP NAME="map1">

<AREA NOHREF SHAPE="circle" COORDS="80,70,40">

<AREA HREF="guide.html" ALT="Путівник" SHAPE="rect"
COORDS="0,0,167,140">

</MAP>

...

<!-- Застосовуємо її до картинки -->

<IMG SRC="*.gif" WIDTH="167" HEIGHT="140" BORDER="0"
ALT="Путівник" USEMAP="#map1">

! У послідовності з декількох елементів AREA найбільший пріоритет мають ті, які визначені першими. Вони перекривають області, що визначені пізніше. Саме тому в наведеному вище прикладі спочатку створюється пасивна область (отвір) і лише потім – активна з посиланням.

! Намагайтеся завжди вказувати атрибут ALT. Користувачі, що працюють у текстовому режимі (а таких чимало!), скажуть вам спасибі.

Спеціальні символи в HTML

Позначення спеціальних символів в HTML 4

Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
		nbsp	 	нерозривний пробіл
¡	¡	isexcl	¡	перевернений знак оклику
¢	¢	cent	¢	цент
£	£	pound	£	фунт стерлінгів
¤	¤	curren	¤	грошова одиниця
¥	¥	yen	¥	їєна або юань
¦	¦	brvbar	¦	розірвана вертикальна риса
§	§	sect	§	параграф
¨	¨	uml	¨	двокрапка – верхній індекс
©	©	copy	©	знак copyright
ª	ª	ordf	ª	feminine ordinal indicator
¬	¬	not	¬	знак заперечення
		shy	­	місце можливого переносу
®	®	reg	®	знак зареєстрованої торговельної марки
–	–	macr	¯	рисочка зверху
°	°	deg	°	градус

Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
±	±	plusmn	±	плюс-мінус
²	²	sup2	²	верхній індекс «два» – «квадрат»
³	³	sup3	³	верхній індекс «три» – «куб»
´	´	acute	´	штрих
µ	µ	micro	µ	мікро
¶	¶	para	¶	недрукований символ
·	·	middot	·	крапка
¸	¸	cedil	¸	кома
¹	¹	sup1	¹	верхній індекс «один»
º	º	ordm	º	masculine ordinal indicator
¼	¼	frac14	¼	дріб – одна чверта
½	½	frac12	½	дріб – одна друга
¾	¾	frac34	¾	дріб – три чверті
÷	÷	divide	÷	знак ділення
ƒ	ƒ	fnof	ƒ	знак функції = florin
Α	Α	Alpha	Α	заголовна альфа
Β	Β	Beta	Β	заголовна бета
Γ	Γ	Gamma	Γ	заголовна гама

Δ	Δ	Delta	Δ	заголовна дельта
Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
Ε	Ε	Epsilon	Ε	заголовна епсилон
Z	Z	Zeta	Ζ	заголовна дзета
Η	Η	Eta	Η	заголовна ета
Θ	Θ	Theta	Θ	заголовна тета
Ι	Ι	Iota	Ι	заголовна йота
Κ	Κ	Kappa	Κ	заголовна капа
Λ	Λ	Lambda	Λ	заголовна лямбда
Μ	Μ	Mu	Μ	заголовна мю
Ν	Ν	Nu	Ν	заголовна ню
Ξ	Ξ	Xi	Ξ	заголовна ксі
Ο	Ο	Omicron	Ο	заголовна омікрон
Π	Π	Pi	Π	заголовна пі
Ρ	Ρ	Rho	Ρ	заголовна ро
Σ	Σ	Sigma	Σ	заголовна сигма
Τ	Τ	Tau	Τ	заголовна тау
Υ	Υ	Upsilon	Υ	заголовна іпсилон
Φ	Φ	Phi	Φ	заголовна фі
Χ	Χ	Chi	Χ	заголовна хі
Ψ	Ψ	Psi	Ψ	заголовна псі
Ω	Ω	Omega	Ω	заголовна омега

α	α	alpha	α	рядкова альфа
Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
β	β	beta	β	рядкова бета
γ	γ	gamma	γ	рядкова гама
δ	δ	delta	δ	рядкова дельта
ε	ε	epsilon	ε	рядкова епсилон
ζ	ζ	zeta	ζ	рядкова дзета
η	η	eta	η	рядкова ета
θ	θ	theta	θ	рядкова тета
ι	ι	iota	ι	рядкова йота
κ	κ	kappa	κ	рядкова капа
λ	λ	lambda	λ	рядкова лямбда
μ	μ	mu	μ	рядкова мю
ν	ν	nu	ν	рядкова ню
ξ	ξ	xi	ξ	рядкова ксі
ο	ο	omicron	ο	рядкова омікрон
π	π	pi	π	рядкова пі
ρ	ρ	rho	ρ	рядкова ро
ς	ς	sigmaf	ς	рядкова сигма (final)
σ	σ	sigma	σ	рядкова сигма
τ	τ	tau	τ	рядкова тау
υ	υ	upsilon	υ	рядкова іпсилон

φ	φ	phi	φ	рядкова фі
Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
χ	χ	chi	χ	рядкова хі
ψ	ψ	psi	ψ	рядкова псі
ω	ω	omega	ω	рядкова омега
•	•	bull	•	малий чорний кружок
...	...	hellip	…	три крапки
'	'	prime	′	одиначний штрих – хвилини і фути
"	"	Prime	″	подвійний штрих – секунди і дюйми
/	/	frac	⁄	скісна дробова риска
℘	℘	weierp	℘	рукописне заголовне латинське Р
ℑ	ℑ	image	ℑ	рукописне заголовне латинське І
℔	℔	real	ℜ	рукописне заголовне латинське R
™	™	trade	™	знак торговельної марки
ℵ	ℵ	alefsym	ℵ	символ алеф – first transfinite cardinal
←	←	larr	←	стрілка вліво
→	→	rarr	→	стрілка вправо
↑	↑	uarr	↑	стрілка нагору

↓	↓	darr	↓	стрілка вниз
Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
↔	↔	harr	↔	стрілка вліво-вправо
↵	↵	crarr	↵	стрілка вниз і вліво – знак повернення каретки
⇑	⇑	uArr	⇑	подвійна стрілка нагору
⇓	⇓	dArr	⇓	подвійна стрілка вниз
⇐	⇐	lArr	⇐	подвійна стрілка вліво
⇒	⇒	rArr	⇒	подвійна стрілка вправо
⇔	⇔	hArr	⇔	подвійна стрілка вліво-вправо
∀	∀	forall	∀	для всіх
∂	∂	part	∂	частковий диференціал
∃	∃	exist	∃	існує
∅	∅	empty	∅	порожня множина; діаметр
∇	∇	nabla	∇	набла = backward difference
∈	∈	isin	∈	належить
∉	∉	notin	∉	не належить

∞	∞	infin	∞	нескінченність
Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
$\ni$	$\ni$	ni	∋	містить
$*$	$*$	lowast	∗	оператор «зірочка»
$\sqrt{}$	$\sqrt{}$	radic	√	квадратний корінь
$\propto$	$\propto$	prop	∝	пропорційно
$\angle$	$\angle$	ang	∠	кут
$\wedge$	$\wedge$	and	∧	логічне І
$\vee$	$\vee$	or	∨	логічне АБО
$\cap$	$\cap$	cap	∩	перетин
$\cup$	$\cup$	cup	∪	об'єднання
$\prod$	$\prod$	prod	∏	добуток
$\sum$	$\sum$	sum	∑	сума
$-$	$-$	minus	−	мінус
$\int$	$\int$	int	∫	інтеграл
$\therefore$	$\therefore$	there4	∴	отже
$\sim$	$\sim$	sim	∼	знак тильда – знак подібності
$\approx$	$\approx$	asymp	≈	майже дорівнює, асимптотично прямує
$\cong$	$\cong$	cong	≅	апроксиматично прямує

Вид по імені	Вид по коду	Ім'я	HTML-код	Опис
$\neq$	$\neq$	ne	≠	не дорівнює
$\equiv$	$\equiv$	equiv	≡	тотожно
$\leq$	$\leq$	le	≤	менше або дорівнює
$\geq$	$\geq$	ge	≥	більше або дорівнює
$\subset$	$\subset$	sub	⊂	є підмножиною
$\supset$	$\supset$	sup	⊃	є надмножиною
$\subseteq$	$\subseteq$	sube	⊆	є підмножиною або дорівнює
$\supseteq$	$\supseteq$	supe	⊇	є надмножиною або дорівнює
$\not\subset$	$\not\subset$	nsup	⊄	не є підмножиною
$\oplus$	$\oplus$	oplus	⊕	плюс у кружку – direct sum
$-$	$-$	ndash	–	коротка рисочка
$—$	$—$	mdash	—	довга рисочка
‘	‘	lsquo	‘	ліві одиночні лапки
’	’	rsquo	’	праві одиночні лапки
,	,	sbquo	‚	нижні одиночні лапки
“	“	ldquo	“	ліві подвійні лапки
”	”	rdquo	”	праві подвійні лапки
„	„	bdquo	„	нижні подвійні лапки
$\dagger$	$\dagger$	dagger	†	кинджал (хрест)

Вид по імені	Вид по коду	Ім'я	HTML Код	Опис
‡	‡	Dagger	‡	подвійний кинджал (хрест)
‰	‰	permil	‰	проміле
‹	‹	lsaquo	‹	ліві кутові одиночні лапки
›	›	rsaquo	›	праві кутові одиночні лапки

Палітра кольорів

У верстці сторінок Ви можете використати як шістнадцятирічний код (RGB), так і назви кольорів. Браузер нормально розпізнає і те, й інше. Нижче перераховані всі статичні кольори (кольори основної палітри) із вказівкою комбінації відповідних RGB-компонентів (у шістнадцятирічному поданні).

R (червоний)	G (зелений)	B (блакитний)	Color для HTML	Кольори в палітрі
00	00	00	black	чорний
80	00	00	darkred	темно-червоний
00	80	00	darkgreen	темно-зелений
80	80	00	darkyellow	темно-жовтий
00	00	80	navy	темно-блакитний
80	00	80	purple	темно-малиновий
00	80	80	darkblue	темно-синій
C0	C0	C0	silver	яскраво-сірий
30	DC	C0	lightgreen	яскраво-зелений
A6	CA	F0	lightcyan	світло-блакитний
FF	FF	F0	ivory	кремовий
D3	D3	D3	lightgrey	яскраво-сірий
80	80	80	grey	сірий
FF	00	00	red	червоний
00	FF	00	green	зелений
FF	FF	00	yellow	жовтий

R (червоний)	G (зелений)	B (блакитний)	Color для HTML	Кольори в палітрі
00	00	FF	blue	синій
FF	00	FF	magenta	малиновий
00	FF	FF	cyan	блакитний (ціан)
FF	FF	FF	white	білий

Більш широкий спектр кольорової палітри можна знайти за такимиаами в Інтернет:

1. <http://www.artlebedev.ru/tools/colors>
2. <http://www.stm.dp.ua/web-design/color-html.php> та ін.

Література

1. Гаевский А. Ю. Информатика 11 класс: Учебное пособие. – 2-е изд., доп. – К.: А.С.К., 2005. – 536 с.
2. Информатика. Базовый курс. Учебник для вузов / Под ред. Симоновича С.В. – СПб: Питер, 2000. – 640 с.
3. Самоучитель работы на ПК / Ковтанюк Ю. С., Соловьян С. В. - К.: Юниор, 2001. – 560 с.
4. Соколов С. А. Java Script в примерах, типовых решениях и задачах. Профессиональная работа. – М.: ООО «И.Д. Вильямс», 2006. – 592 с.
5. Уэплин, Фейт Power Point библия пользователя. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2005. – 768 с.
6. HTML-довідник: <http://html.manual.ru>
7. HTML теги за катеропіями: <http://chebot.ru>
8. HTML-підручник: <http://www.echoecho.com> або <http://www.v-vb.com/>
9. HTML-дизайн: <http://www.artlebedev.ru/kovodstvo>.

<http://www.postroika.ru/books/books5.html>

1. Алексей Гончаров. Самоучитель HTML.
 2. Вильямсон. Универсальный Dynamic HTML.
 3. Владимир Городулин. HTML-справочник.
 4. Дунаев. Самоучитель JavaScript.
 5. Матросов, Сергеев, Чаунин. HTML 4.0.
 6. Морис. HTML в действии.
 7. Николенько. Практические занятия по JavaScript.
 8. Петюшкин. HTML. Экспресс-курс
 9. Ричард Вагнер, Аллен Вайк. JavaScript. Энциклопедия пользователя.
-

Для записів

Для записів

Для записів

Навчальне видання

WEB-дизайн

Навчально-методичний посібник
Для студентів денної форми навчання
спеціальності «Інформатика»

Укладач: **Голуб Євгенія Сергіївна**

Суми: СумДПУ, 2008 р.

Свідоцтво ДК № 231 від 02.11.2000 р.

Відповідальний за випуск **Бугаснко В.В.**

Комп'ютерний набір і верстка **Голуб Є.С.**

Здано в набір 11.02.08. Підписано до друку 11.03.08.
Формат 60x84/16. Гарн. Times New Roman. Друк ризогр.

Папір друк. Умовн. друк. арк. 4,2.

Обл.-вид. арк. 3.7/

Тираж 100 прим. Вид. № 18.

СумДПУ ім. А. С. Макаренка 40002,
40002, м. Суми, вул. Роменська, 87

Виготовлено на обладнанні СумДПУ ім. А.С. Макаренка

Зам. №

74