**МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ**

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Методичні рекомендації
до виконання лабораторних робіт
з навчальної дисципліни
"ТЕХНОЛОГІЇ WEB-ДИЗАЙНУ"

**для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" спеціалізації "Технології електронних мультимедійних видань"
усіх форм навчання**

Харків. Вид. ХНЕУ, 2012

Затверджено на засіданні кафедри комп'ютерних систем і технологій.
Протокол № 2 від 04.09.2011 р.

Укладач Молчанов В. П.

М54 Методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни "Технології WEB-дизайну" для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" спеціалізації "Технології електронних мультимедійних видань" усіх форм навчання / укл. Молчанов В. П. – Х. : Вид. ХНЕУ, 2011. – 64 с. (Укр. мов.)

Подано методичні рекомендації до виконання лабораторних робіт зі створення ресурсів для мережі Інтернет, які містять завдання та основний довідковий матеріал, необхідний для їх виконання.

Рекомендовано для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" спеціалізації "Технології електронних мультимедійних видань" усіх форм навчання.

Вступ

У процесі вивчення навчальної дисципліни "Технології WEB-дизайну" перед студентами стоять такі основні завдання:

отримання цілісного уявлення про процеси функціонування сервісу WWW;

освоєння методики проектування документів для сервісу WWW з урахуванням сучасних дизайнерських концепцій і можливостей технологічних засобів;

придбання уміння використання мов розмітки тексту і засобів програмування для створення повноцінних документів для сервісу WWW;

придбання уміння працювати з сучасними технологічними засобами створення WEB-сторінок і сайтів.

З метою їх виконання до складу дисципліни включені лабораторні роботи, спрямовані на придбання відповідних умінь у рішенні типових задач зі створення ресурсів для мережі Інтернет. Для кожної роботи в методичних рекомендаціях сформульовано перелік таких задач, визначений порядок дій і приведений необхідний для виконання довідковий матеріал, а також сформульовані питання для перевірки повноти засвоєння матеріалу.

Виконання робіт припускає самостійну підготовку напередодні роботи, виконання роботи відповідно до завдання і захист з пред'явленням звіту в електронному вигляді.

Підготовка полягає у вивченні теоретичного матеріалу з використанням конспекту лекцій і рекомендованої літератури, створення необхідної кількості допоміжних файлів з мультимедійними компонентами і тому подібне.

Основною формою звіту у більшості робіт є працездатний ресурс для мережі Інтернет, що функціонує на WEB-сервері або автономно. Під час звіту необхідно відповісти на питання викладача, пов'язані з виконанням роботи. Робота, яка не виконана у відведений навчальний час, закінчується під час самостійної роботи і захищається на наступному занятті.

Змістовний модуль 1. Створення статичних WEB-сторінок

Тема 1. Проектування WEB-сайта

Лабораторна робота 1. Проектування WEB-сайта

Мета роботи – вивчення змісту етапів розробки WEB-ресурсів.

Дане лабораторне заняття забезпечує напрацювання ***уміння*** планувати й реалізовувати заходи щодо створення документів для мережі Інтернет.

Указані вміння надають можливість вирішення таких ***завдань***:

створення плану з розробки WEB-сайта;

обґрунтування вибору основних рішень зі створення WEB-сайтів.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу та продумати тему своєї розробки.

2. Повторити матеріал лекції із стилів WEB-дизайну.

При виконанні лабораторної роботи:

1. Сформулювати мету і завдання своєї публікації.

2. Сформулювати специфічні риси майбутніх відвідувачів, які враховуватимуться при проектуванні.

3. Обґрунтувати структуру майбутнього сайта, сформулювати чинники, що впливають на вибір структури.

4. Сформулювати загальну дизайнерську концепцію, вибрати стиль.

5. Підготувати модульну сітку, ескізи сторінок (схеми розміщення елементів).

6. Підготувати план контента.

7. Розробити концепцію створення сайта.

Звіт з лабораторної роботи представляється у вигляді концепції сайта, що містить опис рішень, прийнятих при виконанні пунктів 1 ─ 6.

Контрольні запитання:

1. Сформулюйте напрями розширення сфери використання мережі Інтернет.

2. Які технології використовуються для надання сторінкам динамічних властивостей?

3. Що таке RSS?

4. Знання яких програмних засобів і для чого може знадобиться творцеві WEB-документів?

5. Опишіть процес взаємодії сервера і браузера.

6. Що таке юзабіліті?

7. Сформулюйте зміст етапів з реалізації проекту створення WEB-сайта.

Довідкові матеріали до лабораторної роботи

Концепція сайта – це документ, розробка якого складає перший крок в реалізації проекту створення сайта. У ньому відображається наступне питання.

Формулюється мета створення, зазвичай за участю замовника, виходячи з його стратегії. Наприклад, мета може бути сформульована так: забезпечити доступ користувачам мережі Інтернет до основних електронних інформаційних ресурсів (зробити діяльність більш відкритою, сформувати імідж) підприємства або організації.

Формулюються список завдань, вирішення яких пристворенні сайта забезпечить досягнення мети. Завданнями можуть бути: розповсюдження інформації, збір інформації, публікація матеріалів, доступ до літератури, надання сервісів і тому подібне

Визначається тип створюваного ресурсу (персональний, інформаційно-довідковий, рекламний, розважальний, освітній, навчальний і тому подібне).

Визначаються характеристики потенційних відвідувачів, особливості яких доцільно врахувати при розробці (особливості поведінки, інтереси і т. д.).

Визначається передбачуваний тип доступу, наприклад, загальний, розмежування на зареєстрованих і незареєстрованих відвідувачів, авторизований, корпоративний і тому подібне.

Визначається склад функцій, який буде доступний відвідувачам. Такими функціями, наприклад, можуть бути:

надання доступу до інформації, документації, до інформаційних баз даних;

публікація новин і оголошень;

створення електронних поштових повідомлень за наданими адресами;

зворотний зв'язок для проведення і збору відгуків;

збір замовлень (інформації про замовлення);

реєстрація відвідувачів з метою збору статистики;

форуми для обговорень;

різні сервіси (переклад текстів, розрахунки) і так далі.

Визначається передбачувана структура сайта, склад основних розділів (сторінок) і їх ієрархія. Структура може бути описана в термінах рубрик або приведена у вигляді схеми.

Обирається стилістичний напрям дизайну і особливості дизайну окремих сторінок. Наприклад, оригінальний для головної сторінки, і єдиний у вибраному стилі для початкових сторінок рубрик, можуть бути визначені особливості сторінок кожного розділу.

Визначається типова структура (модульна сітка) головної сторінки і початкових сторінок рубрик, а також зразковий контент (інформаційне наповнення). При необхідності можуть додаватися ескізи сторінок.

Для отримання повного уявлення про розробку концепції рекомендується ознайомитися результатами пошуку в мережі Інтернет за такими запитами "бриф сайта", "концепція сайта", "технічне завдання на створення сайта" українською та російською мовами.

Література: основна [1]; додаткова [3; 4].

Тема 2. Розмітка тексту з використанням HTML

Лабораторна робота 2. Розміщення текстової інформації на WEB-сторінках

Мета роботи – вивчення мови HTML, прийомів розмітки тексту і створення простих WEB-сторінок.

Дане лабораторне заняття забезпечує напрацювання ***умінь*** створювати WEB-сторінки з використанням мови розмітки тексту HTML.

Указані вміння надають можливість вирішення таких ***завдань***:

створення WEB-сторінок з однім вікном та фреймами;

розміщення на WEB-сторінках текстової інформації та зображень;

форматування текстової інформації за допомогою тегів;

позиціонування елементів на сторінці за допомогою таблиць.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати зміст і підготувати ескізи сторінок для створення під час роботи.

3. Створити необхідну кількість малюнків для фону та елементів оформлення.

При виконанні лабораторної роботи:

1. Спільна робота з додатками Блокнот і Internet Explorer.

1.1. Створити сторінку з фоном і кількома довільними елементами.

1.2. Створити сторінку з фреймами, в кожному з яких відображається своя сторінка.

1.3. Доповнити сторінки мета-тегами, що містять інформацію про автора, ключових словах та кодування. Випробувати мета-теги для переадресації сторінок і установки кодування.

2. Розміщення форматованого тексту.

2.1. Створити сторінку, текст якої форматован різними тегами (теги фізичного і логічного форматування).

2.2. Створити сторінку, що містить структурований з використанням різних засобів текст (списки, тег PRE та інші прийоми), випробувати підстановки.

3. Розміщування на сторінках зображень.

3.1. Вивчити і випробувати різні засоби створення фону сторінки (малюнок, текстура, кольоровий фон).

3.2. Розмістити на сторінках ілюстрації, задати різну взаємодію з текстом, вивчити різні режими відображення зображень.

4. Використовування таблиць.

4.1. Створити таблицю з об'єднаними осередками, що містить різний зміст.

4.2. Навчитися використовувати теги <THEAD>, <TFOOT>, <TBODY>, <COL> І <COLGROUP> для форматування таблиць.

4.3. Вивчити використання таблиць для позиціонування елементів. Створити модульну сітку для сторінок на основі таблиці. Застосувати створену сітку для сторінки, розмістити в кожній області свій фон або малюнок.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Що таке гіпертекстовий документ?

2. Перерахувати основні синтаксичні конструкції мови HTML.

3. Дати визначення понять: тег, елемент, контейнер.

4. Записати приклад використання тегів, що визначають структуру документа.

5. У якому з тегів (який відкриває або закриває) можуть задаватися атрибути?

6. Назвати засоби, що дозволяють змінювати розміри і шрифти написів, що виводяться на сторінці.

7. Назвати види списків, які можуть бути присутніми на сторінці. Записати приклад.

8. Перерахувати основні атрибути тегу <IMG>.

9. У яких тегах може бути присутнім URL файла з малюнком? Записати приклад.

10. Як задати фоновий малюнок на сторінці?

Довідкові матеріали до лабораторної роботи

До пункту 1.

При спільній роботі з додатками Блокнот і Internet Explorer рекомендується така послідовність дій:

- відкрити Блокнот, ввести HTML-код сторінки;
- зберегти у відповідній папці з необхідним ім'ям і розширенням htm (для початкової сторінки це index.htm), вікно Блокнота не закривати;
- відкрити браузер;
- через меню Файл-Відкрити знайти і відкрити сторінку;
- оцінити вигляд сторінки, прийняти рішення про необхідні зміни;
- перемкнутися в Блокнот, внести зміни в код, виконати команду Зберегти;
- перемкнутися в браузер, виконати команду Оновити.

Три останні пункти виконувати до одержання необхідного результату.

При створенні сторінок використовувати атрибути тега BODY (табл. 1), мета-теги та теги створення фреймів (табл. 2).

Таблиця 1

Призначення атрибутів тегу BODY

Ім'я атрибута	Значення	Призначення
background	"шлях до файла"	Створення фонового зображення
bgcolor	#RRGGBB	Колір фону
text	#RRGGBB	Колір тексту
link	#RRGGBB	Колір посилань
vlink	#RRGGBB	Колір використовуваних посилань
alink	#RRGGBB	Колір останньої посилання

Таблиця 2

Теги завдання структури

Ім'я тега	Вміст тега
<HTML>.</HTML>	Контейнер гіпертекстового документа
<HEAD>.</HEAD>	Контейнер заголовка документа
<BODY>.</BODY>	Контейнер тіла документа
<TITLE>.</TITLE>	Контейнер назви документа
<FRAMESET>. </FRAMESET>	Контейнер для фреймів
<FRAME>	Фрейм
<IFRAME>...</IFRAME>	Плаваючий фрейм

До пункту 2.

Основні теги, яки можуть містити текст, наведені у табл. 3, теги логічного та фізичного форматування у табл. 4 та табл. 5.

Таблиця 3

Теги для створення текстових елементів

Ім'я тега	Вміст тега
<P>.</P>	Абзац тексту
<H1>.</H1>. <H6>.</H6>	Заголовки
<UL>.</UL> <OL>.</OL> <DL>.</DL>	Списки
<TABLE>. </TABLE>	Таблиця

Таблиця 4

Теги логічного форматування

Ім'я тега	Призначення
EM	виділення
STRONG	ще сильніше виділення
DFN	означає, що цей термін має визначення
VAR	частина тексту (звичайне слово) є змінною
CITE	назва цитованої книги або статті
BLOCKQUOTE	цитування
CODE	код програми або його еквівалент (наприклад, HTML)
SAMP	службові повідомлення комп'ютера (вивід з програми, скрипти, команди і тому подібне)
KBD	текст, який повинен друкуватися на клавіатурі користувача (зазвичай використовується для інструкцій)
DIV	ділення тексту на розділи

Таблиця 5

Теги фізичного форматування

Ім'я тега	Призначення
TT	"телетайпний" текст, тобто текст одного розміру
I	курсив
B	жирний
U	підкреслення
STRIKE	закреслений текст
BIG	великий шрифт
SMALL	малий шрифт
SUB	підрядковий текст
SUP	надрядковий текст
FONT	установка розміру і кольору шрифту
BASEFONT	базовий розмір шрифту

До пункту 3.

Основні атрибути тега IMG, які впливають на розміщення зображень, наведені у табл. 6.

Таблиця 6

Значення атрибутів тега <IMG>

Атрибут	Значення	Примітка
WIDTH	число	ширина в пікселях
HEIGHT	число	висота в пікселях
ALIGN	BOTTOMMIDDLETOP	При розміщенні зображення в рядку тексту
ALIGN	LEFTRIGHT	При обтіканні зображення текстом
HSPACE	число	Проміжок між текстом і зображенням по горизонталі
VSPACE	число	Проміжок між текстом і зображенням по вертикалі
ALT	текст	Альтернативний текст

До пункту 4.

Основні атрибути тега TABLE, які впливають на зовнішній вигляд таблиць, наведені у табл. 7.

Таблиця 7

Призначення атрибутів, що визначають вид таблиці

Атрибут	Елемент	Призначення
ALIGN	Таблиця, заголовок, рядок, осередок	Вирівнювання таблиці по горизонталі; вирівнювання даних по горизонталі; розміщення заголовка над або під таблицею
VALIGN	Рядок, осередок	Вирівнювання по вертикалі
WIDTH	Таблиця, осередок	Ширина
HEIGHT	Осередок	Висота
COLSPAN	Осередок	Протяжність в декілька стовпців
ROWSPAN	Осередок	Протяжність в декілька рядків
BGCOLOR	Таблиця, осередок	Колір фону
CELLSPACING	Таблиця	Зазор між осередками
CELLPADDING	Таблиця	Зазор між вмістом осередку і її межею
BORDER	Таблиця, осередок	Відображення меж осередків і зовнішньої рамки таблиці

Література: основна [1]; додаткова [3; 4].

Лабораторна робота 3. Створення зв'язаних WEB-сторінок

Мета роботи – вивчення засобів створення елементів навігації, прийомів розмітки тексту і створення зв'язаних за допомогою гіперпосилань WEB-сторінок.

Дане лабораторне заняття забезпечує напрацювання ***умінь*** створювати WEB-сторінки та сайти з використанням мови розмітки HTML.

Указані вміння надають можливість вирішення таких ***завдань***:

створення гіперпосилань на різні ресурси WEB-сайта;

створення елементів навігації різними засобами та властивостями.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати структуру сайта, ескізи сторінок і заготівку сторінки з модульною сіткою для розміщення елементів навігації.

3. Створити необхідну кількість малюнків для карти посилань, альбому, кнопок і елементів оформлення.

При виконанні лабораторної роботи:

1. Створення посилань.

1.1. Створити і випробувати посилання текстом і зображеннями на WEB-сторінки і ресурси інших типів.

1.2. Створити шаблон електронної книжки з оригінальними елементами оформлення (малюнки, маркери і так далі).

1.3. Створити альбом фотографій, що реалізовує оригінальну структуру.

2. Створити карту посилань за власним ескізом.

3. Дослідження базування.

3.1. Створити сайт з використанням абсолютних посилань, випробувати перенесення в інше місце.

3.2. Створити сайт з використанням відносних посилань, задати базу, виконати на нове місце.

4. Створити форму з елементами за власним задумом.

Звіт з лабораторної роботи представляється у вигляді сайта, який містить початкову сторінку з відомостями про виконавця та завдання роботи, та пов'язана зі сторінками, які створені при виконанні кожного пункту. Сайт повинен мати оригінальне оформлення.

Контрольні запитання:

1. Що відбудеться при виконанні клацання на посиланні:
<A HREF="nsck.asp"> ТУТ,
<A HREF="md.mdb"> ТУТ,
<A HREF="nsck.php"> ТУТ
2. Чи можна помістити в тег <A>...</A> текст і малюнок одночасно? Якщо так, то що буде посиланням?
3. Чи можна використовувати посилання для переходу усередині сторінки? Якщо так, то як?
4. Де виконуватиметься пошук ресурсу при клацанні на посиланні <A HREF="..Pg.html"> Перехід?
5. Файли яких форматів можна використовувати при створенні карти посилань?

6. Що необхідно зробити, щоб браузер відображав всі пропуски і табуляцію?

7. Для чого призначені атрибути ISMAP і USEMAP в тегі IMG?

8. Сформулюйте призначення тега MAP, назвіть основні атрибути.

9. Чи можна використовувати теги <input type=...> зовні тега FORM? У чому різниця?

10. Як використовується значення атрибутів value в тегові <SELECT> <OPTION value=1>...</SELECT> ?

Довідкові матеріали до лабораторної роботи

До пункту 1.

Гіперпосилання на різни ресурси мають такий формат:

<A HREF="URL" TARGET="ім'я" TITLE="текст"><P>...</p;

<A HREF="URL" TARGET="ім'я" TITLE="текст"> <IMG...>,

де URL – адреса запрошуваного ресурсу;

ім'я – ім'я вікна, в якому потрібно відкрити ресурс (можна також використовувати спеціальні значення _self – відкрити усередині поточного вікна; _**parent** або _**top** відкрити у вікні без фреймів; _**blank** або **new** – відкрити в новому вікні);

текст – текст, який буде показаний при наведенні курсору.

Шаблон для переходів по сторінці:

```
<UL>
<LI><A HREF="#gl1">ГЛАВА 1</A>
<LI><A HREF="#gl2">ГЛАВА 2</A>
<LI><A HREF="#gl3">ГЛАВА 3</A>
</UL>
.
<A NANE="gl1"></A>
.
<A NANE="gl2"></A>
.
<A NANE="gl3"></A>
```

До пункту 2.

Рис. 1, а точніше його частини (коло та трикутник) можуть бути посиланнями за допомогою наступного коду.

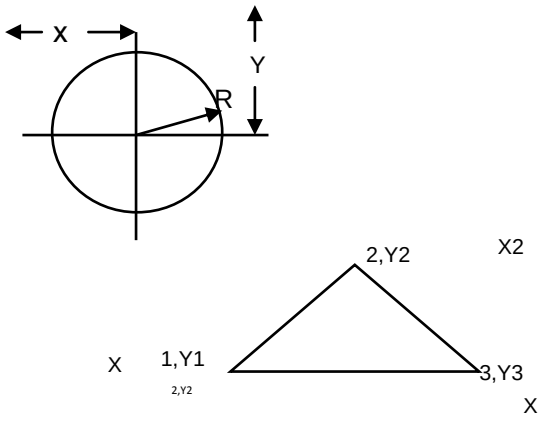


Рис. 1. **Мапа посилань (map.gif)**

Для використання коду потрібно змінні X,Y,R,X1 і т. д. замінити на відповідні значення.

```
<IMG SRC="map.gif" BORDER=0 USEMAP="#map1">
<MAP NAME="map1">
<AREA HREF="primer1.htm" ALT="Текст перший" SHAPE=CIRCLE
COORDS="X,Y,R">
<AREA HREF="primer2.htm" ALT="Текст другий" SHAPE=POLY
COORDS="X1,Y1,X2,Y2,X3,Y3">
</map>
```

До пункту 3.

Зазвичай базовою адресою вважається та, де розташована початкова сторінка сайта. Шлях до цієї папки додається до всіх відносних посилань. Однак при необхідності вона може бути змінена за допомогою тега BASE с атрибутом HREF. Можна використати тільки один тег на сторінку.

Приклади адрес:

абсолютна http://www.myst/f1/f2/p2.htm;

відносна p4.htm - розташована у папці разом з початковою сторінкою;

відносна ../p1.htm – розташована у папці на рівень вище;

відносна ../f2/p2.htm – розташована у іншій папці на тому ж рівні;

відносна f4/p4.htm – розташована у папці на рівень нижче.

До пункту 4

Форми створюються за допомогою тегів FORM (табл. 8), INPUT (табл. 9) та SELECT.

Таблиця 8

Основні атрибути тега FORM

Ім'я атрибута	Значення	Призначення
action	URL або адреса E-mail	указує на ресурс, який буде оброблювати форму
method	GET або POST, за умовчанням – GET	метод пересилки вмісту
name	рядок символів	ім'я елемента

Таблиця 9

Атрибути тега INPUT

Ім'я атрибуту	Значення	Призначення
type	text	однорядкове текстове поле (за умовчанням)
	password	поле введення пароля
	checkbox	перемикач (квадрат, що позначається)
	radio	перемикач (радіокнопка)
	submit	кнопка для відправки даних
	reset	кнопка для приведення полів в початковий стан
	button	кнопка для генерації події
size	число	розмір поля в символах
name	рядок символів	ім'я елемента
value	рядок символів	напис, що виводиться при початковому завантаженні

Зразок тега SELECT:

```
<SELECT name="evaluation">
    <OPTION value=1>...
    <OPTION value=2>...
    <OPTION value=3>...
    <OPTION value=4>...
    <OPTION value=5>...
</SELECT>
```

Література: основна [1]; додаткова [3; 4].

Лабораторна робота 4. Дослідження сторінок складної структури

Мета роботи – вивчення засобів створення елементів навігації при роботі з вікнами, прийомів розмітки тексту на основі модульних сіток і використання мультимедійних об'єктів на WEB-сторінках.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати WEB-сторінки з використанням мов розмітки HTML;

розміщувати на WEB-сторінках різні мультимедійні об'єкти.

Указані вміння надають можливість вирішення таких ***завдань***:

створення сторінок з додатковими вікнами та навігаційними елементами для керування відкриттям сторінок у різних вікнах;

розміщувати на сторінках звукові об'єкти;

розміщувати на сторінках відеооб'єкти;

розміщувати на сторінках об'єкти flash.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати структуру сайта, модульну сітку, ескіз головної сторінки і заготівки інформаційних сторінок.

3. Підготувати невеликі звукові файли форматів wav, mp3.

4. Підготувати невеликі відео файли форматів mp4, wmv, avi, mpeg.

5. Підготувати флеш-фільм.

При виконанні лабораторної роботи:

1. Дослідження навігації по сторінках з вікнами.

1.1. Створити сторінку, що містить вікно на основі плаваючого фрейма.

1.2. Створити сторінку, що містить вікно на основі тега <OBJECT>.

1.3. Порівняти можливості двох способів створення вікон (зовнішній вигляд, управління розміщенням, відображення даних, виконання переходів з використанням елементів навігації на сторінках і кнопок браузера (вперед, назад, відновити).

2. Дослідження засобів вбудовування звукових об'єктів.

2.1. Створити сторінку на основі модульної сітки, з логотипом на основі флеш і посиланнями на інші сторінки. Забезпечити сторінку фоновим звуком.

2.2. Створити сторінку з об'єктами, що програють звукові файли різних форматів (використовувати теги <EMBED> і <OBJECT>).

2.3. Створити сторінку, об'єднавши теги <EMBED> і <OBJECT>, визначити який з них працює.

2.4. Досліджувати відображення сайта в різних браузерах і розмістити власні на окремій сторінці.

3. Дослідження засобів вбудовування відеооб'єктів.

3.1. Створити сторінку з об'єктами, що програють відео файли різних форматів (використовувати і <EMBED> і <OBJECT>).

3.2. Створити сторінку, об'єднавши теги <EMBED> і <OBJECT>, визначити який з них працює.

3.3. Створити сторінку з послідовним заміщенням невідтворних об'єктів на простіші (замість фільму – статичний малюнок)

3.4. Досліджувати відображення сайта в різних браузерах і розмістити власні висновки на окремій сторінці.

Звіт з лабораторної роботи представляється у вигляді сайта, який містить початкову сторінку з відомостями про виконавця та завдання роботи, та пов'язана зі сторінками, які створені при виконанні кожного пункту. Сайт повинен мати оригінальне оформлення.

Контрольні запитання:

1. Що таке об'єкт, який розміщується на WEB-сторінці?
2. Назвіть теги, що дозволяють відображати об'єкти.
3. Поясніть можливості і відмінності у використанні тегів <OBJECT> і <EMBED>.
4. Що таке плагін, елемент ACTIVEX?
5. Поясніть призначення атрибутів **classid, type, data** тега <OBJECT>.
6. Для чого потрібний тег <PARAM>, які атрибути він містить?
7. Що таке фоновий звук на WEB-сторінці? Запишіть тег для створення фонового звуку.
8. Чи можна створити фоновий звук тегом <EMBED>? Якщо так, то як?
9. У чому різниця між потоковим аудіо форматом і звичайним? Назвіть відповідні формати.
10. За допомогою яких тегів можна вбудувати флеш фільм в WEB-сторінку?
11. У чому відмінність відтворення відеофайла через посилання на відповідний ресурс і з використанням тега <OBJECT>?

Довідкові матеріали до лабораторної роботи

До пункту 1.

Плаваючий фрейм створюється за допомогою тега

```
<IFRAME SRC="..."  width="..." height ="..." scrolling={YES,NO,AUTO}
frameborder={1,0}>
альтернативний текст
</IFRAME>
```

Вікно на сторінці може бути створено за допомогою тега:

```
<OBJECT data="..." type="text/html" width="..." height="...">
</OBJECT>
```

До пункту 2.

Фоновий звук можна створити використовуючи теги **<BGSOUND SRC="..." loop="...">**, **<EMBED SRC="..." HIDDEN AUTOSTART="..." LOOP="...">** або **<OBJECT ...>...</OBJECT>**.

Варіанти використання тегів для вбудовування звуку:

```
<object  classid="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
          width="..." height="..."
          type="application/x-oleobject">
 <param name="FileName" value="...">
 </object>

<OBJECT TYPE="audio/mpeg">
 <param name="FileName" value="...">
    альтернативний текст
</object>

<EMBED SRC="..." WIDTH="..." HEIGHT="..." autostart="..."  LOOP="..." >
```

Варіанти об'єднання тегів:

```
<object  classid="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
          width="..." height="..."
          type="application/x-oleobject">
 <param name="FileName" value="...">
 <EMBED SRC="..." AUTOSTART="..." LOOP="...">
 </object>
```

До пункту 3.

Тег для вбудовування флеш:

```
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.ca
b#version=6,0,0,0"  WIDTH="..." HEIGHT="..." id="..." ALIGN="...">
 <PARAM NAME=movie VALUE="...">
 <PARAM NAME=quality VALUE=high>
 <PARAM NAME=bgcolor VALUE=#0000CC>
 <EMBED src="..." quality=high bgcolor=#0000CC  WIDTH="200" HEIGHT="200"
NAME="..." ALIGN="..." TYPE="application/x-shockwave-flash"
PLUGINSPAGE="http://www.macromedia.com/go/getflashplayer">
 </EMBED>
</OBJECT>
```

Варіанти використання тегів для вбудовування відео:

```
<object type="video/mpeg"  width="..." height="...">
```

```
<param name="FileName" value="...">
</object>

<object id="MediaPlayer" width="..." height="..."
classid="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
codebase="http://activex.microsoft.com/activex/controls/mplayer/en/nsmp2inf.cab#
Version=6,4,5,715"  standby="..." type="application/x-oleobject">
   <param name="FileName" value="...">
  </object>

<embed type="application/x-mplayer2" name="MediaPlayer"
   src="..."  width="..." height="..." autostart="..." showdisplay="..."
   howcontrols="..." pluginspage= "….">
```

Література: основна [1]; додаткова [3; 4].

Тема 3. Використання стильових специфікацій

Лабораторна робота 5. Форматування сторінок з використанням таблиць стилів

Мета роботи – вивчення властивостей елементів і засобів форматування WEB-сторінок з використанням різних варіантів таблиць стилів.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

виконувати форматування WEB-сторінок з використанням таблиць стилів;

придати документу потрібній вид за допомогою декількох таблиць CSS.

Указані вміння надають можливість вирішення таких ***завдань***:

планувати й реалізовувати сумісне використання декількох таблиць різного рівня;

позиціонувати елементи сторінки за допомогою CSS;

форматувати текст;

використовувати різни селектори для створення гнучкої системи правил.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати зміст і підготувати ескізи сторінок для створення під час роботи.

3. Створити необхідну кількість елементів для оформлення сторінок.

При виконанні лабораторної роботи:

1. Дослідження особливостей сумісного використання декількох таблиць стилів.

1.1. Створити різні таблиці стилів (у зовнішньому файлі, у тегові STYLE, у атрибуті STYLE) з однаковими властивостями та різними значеннями. Дослідити їх взаємодію, результати дослідження звести в таблицю.

1.2. Порівняти взаємодію зовнішніх таблиць, які підключаються за допомогою тега LINK, директиви @import та користувачем у браузері. Результати дослідження звести в таблицю.

1.3. Випробувати властивість !important для зміни ваги визначень.

2. Вивчення властивостей блокових елементів.

2.1. Створити декілька блокових елементів з різними параметрами області розміщення, підібравши їх так, щоб візуально оцінити кожен параметр і взаємне розташування елементів.

2.2. Позиціонувати п'ять блокових елементів (по кутах вікна і в центрі), досліджувати дії різних браузерів при зміні розмірів вікна і вмісту при різних значеннях display, visibility, overflow, position.

3. Вивчення властивостей рядкових елементів.

3.1. Створити елемент SPAN з властивістю display:inline, випробувати розміщення в ньому різних елементів (текст, зображення).

3.2. Випробувати для SPAN різні значення властивості display, сформулювати основні відмінності в розміщенні.

4. Вивчення властивостей тексту.

4.1. Створити власний оригінальний стиль для абзацу і заголовка другого рівня, використовуючи властивості групи FONT.

4.2. Створити власний оригінальний стиль для абзацу і заголовка третього рівня, використовуючи властивості групи TEXT.

4.3. Створити правило для абзацу з буквицею (виділення першої в абзаці).

5. Вивчення використання різних селекторів. Створити таблицю стилів, що містить набір правив на основі селекторів всіх типів і їх комбінацій, випробувати дію.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням. Висновки та таблиці з результатами досліджень розташувати на окремій сторінці.

Контрольні запитання:

1. Що таке селектор?
2. Сформулюйте рекомендації по використанню селекторів CLASS і ID.
3. Що таке правило?
4. Сформулюйте основні вимоги синтаксису таблиць стилів.
5. У чому полягає основна властивість блокових елементів?
6. Поясніть призначення властивості float. Яких значень воно може набувати?
7. Що таке узагальнена властивість? Як записати його значення?
8. Сформулюйте рекомендації з використання таблиць стилів. У яких випадках які використовувати (зовнішні, в тегах і так далі)?
9. Чи може сторінка містити декілька тегів STYLE?
10. Поясніть, який текст буде отриманий в результаті застосування правила P { font: bold italic large Palatino, serif }? Запишіть це правило у вигляді набору значень для окремих властивостей.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Для встановлення зв'язку із зовнішньою таблицею використовується тег LINK:

<LINK REL=STYLESHEET TYPE="text/css"
HREF="http://www.myserver.com/mysheet.css**">.**

Для включення таблиці стилів в документ використовується тег

<STYLE>
набір правив
</STYLE>

Для імпорту таблиці стилів використовується така конструкція:

<STYLE TYPE="text/css">
@import url(http://www.myserver.com/mystyle.css**);**
</STYLE>

Для включення стилю в тег використовується атрибут STYLE:

STILE="набор правив"

Для використання користувачем власної таблиці стилів в IE необхідно виконати команду **Властивості оглядача...-Общие-Оформле-**

ние..., включити **Оформляти, використовуючи стиль користувача**, натиснути **Огляд...** і вибрати таблицю стилів.

До пункту 2.

У табл. 10 наведені імена і значення основних властивостей, які описують область розміщення елемента.

Таблиця 10

Властивості області розміщення елемента

Ім'я	Значення	Примітка
Margin Padding Border-width	<число><розмірність > або <число>% або auto	Розмірність – px , in, cm , mm, pc пікселі, дюйми, см, мм, пункти. Наприклад, 25mm; % від базового розміру
Border-color	Black, coral, orange, і так далі або через rgb	#rrggbb (наприклад, #00cc00) rgb(x,x,x), де x=0...255, rgb(x%,x%,x%), де x=0.0...100.0
Border-style	None, dotted, dashed, solid, double, groove, ridge, inset, outset	Різні варіанти рамок: відсутня, пунктирна, штрихова і так далі

До пункту 3.

Рядкові елементі (display= inline) відображається як вбудований в рядок, вміст елементів починається з того місця, де закінчився попередній.

До елементів із значенням list-item можна додавати такі властивості (табл. 11).

Таблиця 11

Таблиця властивостей елементів з display=list-item

Ім'я	Значення	Примітка
list-style	<набір значень>	узагальнена властивість
list-style-image	url ("шлях"), none	зображення маркера
list-style-position	outside, inside	розміщення маркера
list-style-type	disc, circle, square ...	вид маркера

До пункту 4.

При форматуванні тексту можна керувати, як відображенням літер (табл. 12), так і тексту в цілому (табл. 13).

Таблиця 12

Властивості шрифтів

Ім'я	Значення	Примітка
font	<набір значень>	узагальнена властивість
font-family	<список імен>	список імен шрифтів або сімейств
font-size	число, число %, число і ед. изм., xx-small, x-small, small	розмір шрифту
font-style	normal, italic, oblique	зображення шрифту
font-variant	normal, small-caps	представлення рядкових букв
font-weight	normal, bold, bolder, lighter, 100, 200 ...	насиченість шрифту

Таблиця 13

Властивості тексту

Ім'я	Значення	Примітка
letter-spacing	<число>, normal	інтервал між символами
line-height	<число>, <число>%, normal	міжрядковий інтервал
text-align	left, right, center, justify	горизонтальне вирівнювання
text-decoration	none, underline, overline, line-through, blink	варіант оформлення
text-indent	<число>, <число>%	відступ першого рядка
vertical-align	baseline, sub, super, top-text, top, middle, bottom, bottom-text, <число>%	вирівнювання по вертикалі до навколишнього тексту
word-spacing	<число>, normal	інтервал між словами

До пункту 5.

Класовий селектор складається з імені тега й імені класу, з'єднаних крапкою. Наприклад, для тега <H1 CLASS="bl">TEXT</H1> можуть бути:

H1.bl {color:blue; size:20pt}

.bl {color:blue}

ID селектор починається з символу #. Наприклад, для тега <H1 ID="rd">TEXT</H1>:

#rd {color:blue; size:20pt}

У тегові обидва атрибути (CLASS і ID) можуть використовуватися одночасно, довизначаючи стиль елемента.

Контекстні селектори складаються з простих селекторів, розділених пропуском (всі описувані до цього селектори були простими селекторами). Вони застосовуються до елементів, зв'язаних спадковими стосунками і задають властивості тільки конкретного дочірнього елемента.

OL LI {list-style-type: decimal}

UL LI { list-style-type: square}

При використанні таких правил елементи в нумерованому списку (OL) матимуть один стиль, а в ненумерованому (UL) – інший.

Контекстні селектори можуть містити тип елемента, 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 }
```

Псевдокласи розрізняють типи одного елемента (наприклад, посилання в різних станах, активна, вже відвідувалася, ще не відвідувалася), створюючи при визначенні власні стилі для кожного з них. Наприклад, псевдокласи елемента <a href=" "> :

link (посилання), active (активне посилання), visited (відвіданий раніше URL), hover (псевдоклас, що виникає при тому, що піднесло курсору до посилання).

a:link,a:visited {color:blue} – відвіданий раніше URL

a:active {color:red} – активне посилання

a:hover {text-decoration:none} – псевдоклас, що виникає при наведенні курсору до посилання.

Псевдоелементи є частинами інших елементів, задаючи цим частинам відмінний від елемента в цілому стиль (наприклад, перший рядок в абзаці або перша рядка):

P:first-line {color: purple }

H1:first-letter { color: red }.

Література: основна [1]; додаткова [3; 4].

Лабораторна робота 6. Дослідження елементів дизайну, створених на основі таблиць стилів

Мета роботи – вивчення і освоєння засобів і прийомів верстки WEB-сторінок з використанням таблиць стилів.

Дане лабораторне заняття забезпечує напрацювання ***уміння*** використовувати специфікації CSS для верстки WEB-сторінок.

Указані вміння надають можливість вирішення таких ***завдань***:

планувати й реалізовувати сумісне використання декількох таблиць різного рівня;

оформлювати текстові елементи та керувати їх взаємним розташуванням;

верстати сторінку за допомогою по-різному позиціонованих розділів.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати зміст і підготувати ескізи сторінок для створення під час роботи.

3. Підготувати всі необхідні елементи оформлення.

При виконанні лабораторної роботи:

1. Використання фону.

1.1. Створити сторінки з різними варіантами використання фону (розмножити, центрувати, непрокручуваний і так далі).

1.2. Створити правило для абзацу з власним фоном, прозорим фоном.

1.3. Створити правила, що забезпечують різний фон у абзацу і першого рядка

2. Позиціонуванняелементів.

2.1. Створити сторінку здекількома елементами, накладеними один на одного. Випробувати різні значення властивості z-index для керування видимістю елементів.

2.2. Створити сторінку зелементами, що абсолютно позиціонуються, випробувати управління видимістю.

2.3. Створити на основі власного макету дві однотипні сторінки для верстки в дві колонки (одну на основі таблиць, іншу – розділів, що позиціонуються), порівняти результати в різних браузерах при різних розмірах вікна і дозволі.

2.4. Створити елемент для відображення флеш-фільму, розташованийнезалежно від прокрутки.

3. Виконати верстку сторінок для індивідуального завдання.

4. Створити сторінку, використовуючи одне з нововведень CSS3 і HTML5 по власному задуму.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Для чого призначені елементи SPAN і DIV?

2. Чи можна позиціонувати елементи SPAN?

3. Запишіть правило, що забезпечує форматування, еквівалентне тега **B**, тега **I**.

4. Чи може бути присутнім фон у стічних елементів?

5. Назвітьвластивості, що забезпечують управління фоном елементів.

6. Назвіть декілька застосуваньдля управління видимістю елементів.

7. Чи викликає зміну видимості елементів зміна в їх взаємному розташуванні?

8. Які властивості мають значення repeat-y? hidden?

9. Як можна задати вид курсору при наведенні його на елемент?

Довідкові матеріали до лабораторної роботи

До пункту 1.

З кожним елементом пов'язаний фон (таким чином, фон може бути не тільки у всієї сторінки, але і у окремих елементів). Управління фоном забезпечує група властивостей background, вони застосовуються до всіх елементів і не успадковуються. Властивості приведені в табл. 14.

Таблиця 14

Властивості для управління фоном

Ім'я	Значення	Примітка
Background	<набір значень>	Узагальнена властивість
Background-attachment	Fixed, scroll	Режим прокрутки
Background-color	<колір>, transparent	Колір фону
Background-image	Url (шлях до файла), none	Фоновий малюнок
Background-position	<число>%, left, center, right, top, center, bottom	Розташування малюнка
Background-repeat	Repeat-x, repeat-y, repeat, no-repeat	Повторюваність малюнка

До пункту 2.

На розміщення елементів впливають такі властивості (табл. 15).

Таблиця 15

Властивості, використовувані для позиціонування

Ім'я	**Значення**	**Примітка**
display	block, inline, list-item, none	визначає тип елемента
position	absolute, fixed, relative, static	задає систему позиціонування
top, right, left, bottom	<число>	визначають положення елемента
float	right, left, none	позиціонується окремо від потоку елементів
z-index	<число>	управляє порядком накладення елементів
visibility	visible, hidden, collapse	управління видимістю елемента
overflow	auto, hidden, scroll, visible	режим відображення вмісту

До пункту 3.

Приклад верстки у дві колонки:

```
<style>
#main {float:right; width:65%;}
#sections {float:left; width:35%;}
…
</style>
<div id="sections">
...
</div>
<div id="main">
...
</div>
```

Приклад верстки у три колонки:

```
<style>
#main {float:left;width:55%;}
#sections {float:left;width:20%;}
#news {float:right; width:25%;}
…
</style>
<div id="sections">
...
</div>
<div id="main">
...
</div>
<div id=" news ">
…
</div>
```

До пункту 4.

Нові теги (HTML 5): section, article, aside, hgroup, header, footer, nav, dialog, figure, source, mark, progress, meter, time, ruby, rt, rp, canvas, command, detailes, datalist, keygen, output

Нові властивості (CSS 3): border-radius, box-shadow, animation, gradient.

Література: основна [1]; додаткова [3; 4].

Змістовний модуль 2. Клієнтські технології створення динамічних WEB-сторінок

Тема 4. Сценарії, що виконуються на клієнтській стороні

Лабораторна робота 7. Розробка сценаріїв для WEB-сторінок

Мета роботи – вивчення середовища, прийомів створення і відладки сценаріїв для WEB-сторінок.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати динамічні сторінки, використовуючи засоби програмування на стороні клієнта WWW;

виконувати перевірку і відлагодження створюваних програмних елементів.

Указані вміння надають можливість вирішення таких ***завдань***:

розміщувати сценарії на сторінках WEB-сайта;

розробляти та підлагоджувати сценарії з використанням об'єктів.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати алгоритми і тексти сценаріїв, що розробляються.

При виконанні лабораторної роботи:

Розміщення і виконання сценаріїв.

1.1. Розробити сценарій за індивідуальним завданням.

1.2. Випробувати різні варіанти розміщення сценарію (у тегові і у файлі).

2. Досліджувати різні способи перетворення типів.

2.1. Створити сценарій з використанням визначення типів і їх явного перетворення при введенні, виводі і в ході обчислень.

2.2. Створити сценарій з формуванням рядка для виведення даних (один рядок, декілька рядків).

3. Використання операторів, що управляють.

3.1. Розробити сценарій за індивідуальним завданням

3.2. Записати створений сценарій використання різних операторів циклу.

4. Використання вбудованих об'єктів.

4.1. Розробити і випробувати сценарій, що використовує об'єкт Date.

4.2. Розробити і випробувати сценарій з використанням об'єкта String і регулярних виразів.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Які програмні елементи може містити WEB-сторінка?

2. У чому різниця між сценарієм (скриптом) і аплетом?

3. Чи можуть в тегові SCRIPT міститися виконувані оператори? Виклики функцій? Якщо так, то вкажіть момент і число повторень їх виконання.

4. Дані яких типів можна обробляти з використанням скриптів?

5. Що таке блок?

6. Запишіть приклади можливих констант, допустимих в JavaScript.

7. Як визначити тип даних, привласнених змінної?

8. Запишіть приклади різних операторів привласнення.

9. Запишіть формат і поясніть особливості виконання оператора switch.

10. Запишіть і поясніть особливості виконання різних форм оператора while.

Зразки завдань до лабораторної роботи 7

Варіант 1

Ввестимасив зN цілих чисел. Сформувати новий масив, кожен елемент якого дорівнює елемента введеногомасиву, збільшеному на значення, що вводиться з клавіатури.

Розробити і відладити сценарій, що визначає число днів, що залишилися до 23.02.2015.

Розробити і відладити сценарій для перевірки правильності адреси електронній пошти.

Варіант 2

Заповнити масив zN цілих чисел випадковими значеннями. Обчислити суму елементів масиву зпарними номерами.

Розробити і відладити сценарій, що визначає, скільки повних місяців залишилося до 8.03.2015.

Розробити і відладити сценарій для перевірки правильності запису дати у форматі 12.12.2012.

Варіант 3

Заповнити масив випадковими значеннями, визначити число елементів масиву, що потрапляють в інтервал, введений з клавіатури.

Розробити і відладитисценарій, що визначає, скільки повних років залишилося до 8.03.2024.

Розробити і відладитисценарій для перевірки правильності запису про ціну товару у форматі 12 грн 23 коп.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Для розміщення кодів сценаріїв у тексті сторінки служить спеціальний парний тег SCRIPT. Він може містити або текст сценарію, або посилання на файл з розширенням js, що містить сценарій. Сам тег розміщується або в заголовку, або в тілі сторінки, а може і там, і там. Відмінність полягає в доступності імен і можливості їх використання. Рекомендується розміщувати його в заголовку. Крім того, текст сценарію може бути присутнім в інших тегах, наприклад, у посиланнях.

Наприклад,

```
<SCRIPT LANGUAGE="JavaScript ">
// текст сценарію
</SCRIPT >
```

або

```
<SCRIPT LANGUAGE="JavaScript " SRC="root/my.js">
</SCRIPT >
```

Усередині файла my.js тільки текст сценарію і ніяких тегів.

До пункту 2.

У JavaScript тип змінних явно не задається, а визначається за типом привласнюваного значення і може бути динамічно змінений, тобто одній і тій же змінній можна послідовно привласнювати значення різних типів. Підтримується обробка таких типів: цілі, плаваючі, строкові, логічні. При сумісному використанні у вираженнях у більшості випадків виконуються неявні перетворення і приведення до одного типу. Хоча є ряд спеціальних функцій перетворення (parseInt () – перетворення в ціле, parseFloat () – перетворення в таке, що плаває, плаваюче). Під час перетворення в ціле можна вказати підставу (parseInt(x,8) – перетворення у восьмеричне, parseInt(x,16) – перетворення в шістнадцятиричне). Наприклад,

```
result = parseInt ("42") // привласнене ціле значення 42
result = parseInt ("42.33") // привласнене ціле значення 42
result = ("" + 2500) // привласнений рядок "2500"
```

Рядкові змінні є об'єктами і мають властивість length, тому:

```
result = ("" + 2500).length // привласнене ціле значення 4
```

Для визначення типу значення, привласненого змінній, вико ристовується спеціальний оператор **typeof**, який повертає рядок "number" для цілих і таких, що плавають, "string" – для рядкових, "boolean" – для логічних, "undefined" – для помилкових і непроініціалізованих.

До пункту 3.

Оператори, що управляють обчислюванням, схожі на аналогічні у мові С.

Умовний оператор:

```
if (умова){ блок 1 }
else { блок 2 };
```

Оператор-перемикач

```
switch (вираз ) {
case значение1:оператор;
break ;
case значение2:оператор;
break;
...
default:оператор;
}
```

Оператори циклу:

```
for (вираження1 ; умова; вираження2)
{ ... }

while (умова)
{... }

do
{... }
while (умова);
```

До пункту 4.

У мовіJavaScriptкористувач не може безпосередньо створювати власні класи, йому доступні тільки вбудовані класи JavaScript(Date, Math, String, Imageта ін.) і об'єкти браузера.

Об'єкти Datастворюються конструктором var d=newDate(), при використанні конструктора без параметрів створюється об'єкт з поточними на момент виконання програми значеннями часу і дати. За наявності параметрів значення формується на їх основі:

var sDate=new Date("Month dd,yyyy hh:mm:ss")

У об'єкта є декілька методів, наприклад:

sDate.toLocaleString() – вивід в національному форматі

Клас String. Рядки також є об'єктами. Вони створюються при наданні строкового значення змінній або з використанням конструктора varSt=new String("рядок символів");

Клас містить властивість length і безліч методів, з яких найчастіше використовуються cartAt(i), – повертає символ, що стоїть на і-ому місці, indexOf("...",i) – шукає входження підрядка, пошук починається з позиції і, повертає номер позиції початку першого входження.

При роботі з рядками можна використовувати регулярні вирази.

Клас Math. Об'єкти цього класу не вимагають створення, його властивостями є математичні константи, а методами – математичні функції. Не дивлячись на те, що на JavaScript рідко розробляються програми обчислювального характеру, в деяких випадках вони можуть виявитися корисними. У табл. 16 приведені основні математичні функції.

Таблиця 16

Математичні функції

Метод	Опис	Метод	Опис
acos()	арккосинус	ceil()	найближче ціле зверху
asin()	арксинус	floor()	найближче ціле знизу
atan()	арктангенс	round()	найближче ціле
cos()	косинус	max()	максимальний із списку
sin()	синус	min()	мінімальний із списку
exp()	експонента (ex)	sqrt()	корінь квадратний
log()	логарифм натуральний	random()	випадкове число (0...1)

Наприклад, після виконання оператора alert(Math.random()) у вікні діалогу з'явиться псевдовипадкове значення, тобто при кожному повторному виконанні воно буде іншим.

Література: основна [1]; додаткова [2 – 4].

Лабораторна робота 8. Виконання сценаріїв, вбудованих у WEB-сторінки.

Мета роботи – вивчення методів скріплення сценаріїв з подіями різних елементів WEB-сторінки.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати динамічні сторінки, використовуючи засоби програмування на стороні клієнта WWW;

виконувати аналіз та створення обробників подій у складі WEB-сторінки.

Указані вміння надають можливість вирішення таких ***завдань***:

керувати виконанням сценаріїв на сторінках WEB-сайта;

зв’язувати обробники подій з подіями у різних елементах різними засобами.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати алгоритми і тексти сценаріїв, що розробляються.

При виконанні лабораторної роботи:

1. Розміщення сценаріїв

1.1. Розробити сценарій за індивідуальним завданням.

1.2. Випробувати різні варіанти виконання сценаріїв (виконання операторів в ході завантаження і при виклику функцій).

2. Досліджувати різні способи обробників з подіями, сформулювати рекомендації по застосуванню.

2.1. Перевизначити стандартний обробник в посиланні.

2.2. Розмістити в тегові атрибута подій (onClick, onLoad і так далі).

2.3. Призначити обробник за допомогою атрибутів FOR і EVENT.

2.4. Динамічно зв'язати обробник з подією.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Чи можуть в тегові SCRIPT міститися виконувані оператори? Виклики функцій? Якщо так, то вкажіть момент і число повторень їх виконання.

2. Сформулюйте рекомендації з використання атрибутів подій в тегах і атрибутах FOR і EVENT3. Що означає зв'язати обробник з подією динамічно?

4. Чи є у об'єкта Image методи? Якщо є, то назвіть їх. Чи можна використати їх у обробниках подій?

5. Чи допустимий такий вираз dd=Date() - Date();? Якщо так, то якого значення набуде змінна dd?

6. Запишіть різні варіанти доступу до властивостей об'єкта Image? Сформулюйте рекомендації щодо їх використання.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Оператори сценарію, що розміщуються в тегові SCRIPT виконуються, коли браузер інтерпретує вміст тега.

Оператори, розміщені у функціях, виконуються після виклику функцій (після звернення до функції). Виклик відбувається або в тегові (там, де записано звернення), або коли відбувається подія, з якою пов'язана функція (така функція називається обробником події).

До пункту 2.

Пов'язання обробників з подіями може проводиться за замовчуванням (наприклад, для клацання по посиланню). Такі обробники можна перевизначити. Наприклад, <a href="http://www.narod.ru')"> замінити на <a href="javascript:window.alert('Do you speak English?')">.

Пов'язання обробників з подіями може проводиться за допомогою спеціальних атрибутів в тегах. Наприклад:

```
<input type=button value="ТУТ" onClick="foo();">
<BODY onLoad ="main()">
```

або так

```
<script FOR="gener" EVENT="onClick" LANGUAGE="JavaScript">
//текст обробника
</script>
```

Пов'язання обробників з подіями може проводиться також динаміч-но в ході виконання сценарію. Наприклад:

```
<INPUT TYPE=BUTTON ID="myButton" VALUE="Click here">
<SCRIPT LANGUAGE="JavaScript">
  document.all.myButton.onclick =  new Function("alert('Hello');");
</SCRIPT>
```

або так

```
<SCRIPT LANGUAGE="JavaScript">
function clicked ()
{
…
}
…
document.all.myButton.onclick = clicked;
</SCRIPT>
```

Найчастіше використаються такі події (табл. 17).

Таблиця 17

Список часто використовуваних подій

Об'єкт	Подія	Момент виникнення
window	onload	Закінчення завантаження сторінки
	onunload	Відхід із сторінки для завантаження нового
image	onload	Закінчення завантаження зображення
link	onclick	Клацання мишею
	onmouseout	Відведення курсору з посилання
	onmouseover	Наведення курсору на посилання
Елементи форми	onblur	Втрата фокусу
	onfocus	Отримання фокусу
	onselect	Вибір тексту усередині поля text, textarea
	oncange	Зміна значення усередині поля
	onclick	Клацання мишею на кнопці або перемикачі

Література: основна [1]; додаткова [2 – 4].

Тема 5. Створення динамічних сторінок

Лабораторна робота 9. Використання об'єктів DOM в скриптах

Мета роботи – вивчення середовища і прийомів створення динамічних WEB-сторінок.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати динамічні сторінки, використовуючи засоби програмування на стороні клієнта WWW;

використовувати об’єкти DOM у сценаріях на WEB-сторінках.

Указані вміння надають можливість вирішення таких ***завдань***:

створення WEB-сторінок з динамічним вмістом;

створювати інтерактивні елементи та власні елементи інтерфейсу.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати алгоритми і тексти сценаріїв, що розробляються.

При виконанні лабораторної роботи:

1. Вивчити властивості об'єкта відповідно до індивідуального завдання.

2. Розробити динамічну сторінку відповідно до індивідуального завдання.

2.1. Сформувати сторінку через вхідний потік.

2.2. Завантажити сторінку і змінити елемент у відповідь на дії користувача.

3. Розробити динамічну сторінку, що використовує форми і вікна як інтерфейсні елементи (за власним задумом)

3.1. Створити у відповідь на подію вікно з текстом.

3.2. Створити інтерактивну форму (калькулятор, опитувальний лист і тому подібне).

3.3. Створити на сторінці власну систему управління переміщенням завантаженими раніше сторінками.

3.4. Створити на основі форми меню посилань.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Якими способами можна передати значення всередину функції?

2. Запишіть фрагмент програми, що містить глобальні і локальні змінні.

3. Перерахуєте способи обробників з подіями.

4. Поясніть механізм використання подій, пов'язаних з таймером?

Чи можуть в тегові SCRIPT міститися виклики функцій? Якщо так, то вкажіть момент і число повторень їх виконання.

5. Запишіть оператора, використовуючи який можна набути значення поточної дати і часу.

6. За допомогою якого атрибута можна ідентифікувати форму, елемент форми?

7. Запишіть всі можливі способи доступу до об'єкта, властивості якого ви досліджували.

Зразки завдань до лабораторної роботи 9

Варіант 1

Вивести і вивчити властивості об'єкта image.

Розробити функцію для зміни розмірів зображення, розміри зображення вводити в діалозі.

Варіант 2

Вивести і вивчити властивості об'єкта form.

Розробити функцію для зміни тексту в полі форми, текст водити в діалозі.

Варіант 3

Вивести і вивчити властивості об'єкта input.

Розробити функцію для зміни зображення відповідно до введеної адреси.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Для доступу до властивостей об'єкта можна скористатися оператором Використовувати оператора **for (... in ...) ...;**

Наприклад, так:

```
<table border='1' cellpadding='3' cellspasing='0'>
<caption>Властивість об’єкту  document</caption>
<tr>
<td>Властивість</td><td>Значення</td>
</tr>
<script language="JavaScript">

for(props in window.document.location) {
```

```
                document.write('<tr>');
                document.write('<td>'+props+'</td>');

document.write('<td>'+window.document.location[props]+' </td>');
                document.write('</tr>');
            }
</script>
```

До пункту 2.

Метод write("текст") записує текст у вхідний потік браузера в тому місці, в якому викликаний. Він використовується для динамічного формування коду HTML-документа. Код можна згенерувати залежно від деяких умов, наприклад, типу або версії браузера.

До пункту 3.

Вікна для діалогу та елементи інтерфейсу можуть створюватися за допомогою об’єктів window (табл. 18) та form (табл. 19).

Таблиця 18

Властивості і методи об'єкта window

Властивості	
Властивість	Призначення
name	Ім'я вікна або фрейма
defaultstatus	Вміст рядка статусу
length	кількість фреймів
Методи	
Метод	Дія
open()	Створення нового вікна
close()	Закриття вікна
focus()	Передача фокусу у вікно
resizeTo(width, height)	Зміна розмірів вікна

Табл. 19

Властивості і методи об'єктів form

Властивості	
Властивість	Призначення
action	адреса програми на сервері для обробки даних
name	ім'я форми
target	задає вікно для виведення даних, передаваних сервером
elements	посилання на колекцію елементів форми
Методи	
Метод	Дія
submit()	відправка даних на сервер
reset()	очищення елементів форми

Література: основна [1]; додаткова [2 – 4].

Лабораторна робота 10. Створення динамічних сторінок

Мета роботи – вивчення методів і прийомів створення динамічних сторінок і внесення змін в скрипи з метою їх адаптації до конкретних умов.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати динамічні сторінки, використовуючи засоби програмування на стороні клієнта WWW;

аналізувати дію обробників подій у складі WEB-сторінки.

Указані вміння надають можливість вирішення таких ***завдань***:

створення різних ефектів на WEB-сторінках;

аналізувати дію обробників подій.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Підготувати алгоритми і тексти сценаріїв, що розробляються.

При виконанні лабораторної роботи:

1. Створення ефектів, пов'язаних з позиціонуванням і розмірами елементів.

1.1. Створити на сторінці елемент, який можна переміщувати мишею (запропонувати і погодити з викладачем).

1.2. Створити на сторінці елемент, розмірами якого можна управляти (запропонувати і погодити з викладачем).

2. Створення ефектів, пов'язаних з накладенням і видимістю об'єктів.

2.1. Створити сторінку з елементами, що змінюють порядок відображення у відповідь на дії користувача.

2.2. Створити сторінку з випадними меню.

3. Зміна змісту об'єктів.

3.1. Створити сторінку з кодом, адаптованим до деякої умови (за узгодженням з викладачем).

3.2. Створити сторінку, що змінює зміст у відповідь на дії користувача (за узгодженням з викладачем).

4. Розробити і реалізувати власний ефект.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Поясніть термін динамічний HTML.

2. При виконанні якої умови наведений код буде правильним: document .all("im ").width=document.all("im ").width*2;?

3. Як записується властивість z-index в скриптах?

4. Які методи використовуються для прив'язки викликів функцій до таймера?

5. Які дії виконують методи window .open() і id .document.open()?

6. На що указує таке посилання: document.all("im ").all?

7. Чи є різниця між значеннями властивостей innerText і outerText ? Якщо так, то в чому?

Довідкові матеріали до лабораторної роботи

До пункту 1.

Браузер виконує позиціонування елементів відповідно до місця тегів у вхідному потоці і властивостей елементів, які задаються атрибутами і правилами з таблиць стилів. Атрибути, що впливають на розміщення: WIDTH, HEIGHT, ALIGN.

Основні властивості стилів, які пов’язані з розміщенням елементів, наведені в табл. 15 (лабораторна робота 6).

До пункту 2.

Для управління видимістю елементів (створення випадних меню, зміна одного елемента іншим) використовуються властивості об'єкта style visibility і zIndex.

До пункту 3.

Зміна вмісту сторінки залежно від умов, що перевіряються, або дій користувача може проводитися в ході завантаження з використанням методу write() або після закінчення шляхом зміни відповідних властивостей елементів (табл. 20).

Таблиця 20

Властивості та методи елементів для зміни змісту

Властивості	
innerHTML	вміст елемента (текст і теги дочірніх елементів)
innerText	текстовий вміст елемента (без тегів)
outerHTML	вміст елемента (включаючи теги)
outerText	текстовий вміст елемента (без тегів)
tagName	тег елемента (без кутових дужок)
Методи	
insertAdjacentHTML("місце","new text"), місце={BeforeBegin,AfterBegin,BeforeEnd,AfterEnd}	вставка тексту з тегами всередину елемента
insertAdjacentText("місце","new text"), місце={BeforeBegin,AfterBegin,BeforeEnd,AfterEnd}	вставка тексту всередину елемента

Література: основна [1]; додаткова [2 – 4].

Змістовний модуль 3. Серверні технології створення динамічних WEB-сторінок

Тема 6. Характеристика серверних технологій

Лабораторна робота 11. Дослідження взаємодії браузера з сервером на основі CGI

Мета роботи – вивчення можливостей створення динамічних сторінок з використанням CGI-програм та придбання практичних навиків програмування на боці сервера.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

обґрунтовувати вибір потрібної технології створення WEB-додатків;

створювати динамічні сторінки і обробляти дані з форм, вико ристовуючи засоби програмування на стороні сервера WWW.

Указані вміння надають можливість вирішення таких ***завдань***:

налаштовувати WEB-сервери для роботи зі сторінками, які формуються динамічно;

створювати CGI-програми у середовищі Visual Studio;

обробляти дані з форм за допомогою CGI-програм.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати тему своєї розробки, підготувати необхідні алгоритми і програми.

При виконанні лабораторної роботи:

1. Ознайомитися з налаштуваннями сервера IIS.

2. Створити консольний додаток у середовищі VS для видачі клієнтові простої WEB-сторінки.

3. Випробувати роботу створеної CGI-програми.

4. Створити CGI-програму для обробки даних з форми за індивідуальним завданням.

Звіт з лабораторної роботи надається у вигляді продукту, який створено при виконанні роботи за кожним пунктом завдання та оформлено за особистим задумом.

Контрольні запитання:

1. Який з підходів до створення динамічних сторінок представляється вам найбільш універсальним? Чому?

2. Який з підходів до створення динамічних сторінок представляється вам найбільш простим в реалізації? Чому?

3. Поясните зв'язок між розширеннями сервера і технологіями, заснованими на скриптах, вбудованих в сторінку.

4. Дайте характеристику інтерфейсу CGI?

5. У чому відмінність технологій CGI і ISAPI?

6. Як здійснюється доступ до змінних оточення в середовищі VS?

7. Чим відрізняється передача даних при використанні методів GET і POST?

Зразки завдань до лабораторної роботи 11

Варіант 1

Створити сторінку для заповнення анкети нового користувача.

Варіант 2

Створити сторінку для входу на сайт з перевіркою імені і пароля.

Варіант 3

Створити сторінку для збору статистики за віком відвідувачів

Довідкові матеріали до лабораторної роботи

До пункту 1.

Для тестування динамічних сторінок на комп'ютері має бути встановлений WEB-сервер. При використанні операційної системи Windows XP найпростішим рішенням є установка Internet Information Services(IIS). **Панель керування ─ Установка і видалення програм ─ Установка компонентів Windows**, запуститься майстер, зазначити перемикач **Internet Information Services(IIS)** і виконати установку.

Налаштування сервера проводяться через оснащення Internet Information Services, ярлик якого розміщений на вкладці Адміністрування (**Панель керування ─ Адміністрування**).

Для розміщення сайта з динамічними сторінками необхідно створити віртуальну папку, яка зв'яже ім'я сайта на сервері з однією з фізичних папок комп'ютера. Папка створюється в дереві папок **WEB-вузол за замовчування** по команді **Створити ─ Віртуальний каталог** з контекстного меню. Запуститься майстер, в діалозі з яким будуть задані всі необхідні імена і права доступу. Надалі доступ до налаштувань віртуальної папки здійснюється по команді **Властивості** з контекстного меню. Необхідно задати дозволені дії з каталогом, дозвіл на запуск сценаріїв і виконуваних файлів, можна виконати й інші налаштування

(наприклад задати імена, використовувані за умовчанням та ін.). Після виконання налаштувань для доступу до розміщених ресурсів використовується адреса **http:///localhost/им'я_вір_папки/им'я_ресурсу**.

До пункту 2.

Запустити VS, створити новий консольний додаток, в заготівку функції Main вставити текст

```
Console.Write("Content-Type:text/html\n\n");
Console.WriteLine("<html>");
Console.WriteLine("<head>");
Console.WriteLine("<title>Простійша CGI-програма</title>");
Console.WriteLine("</head>");
Console.WriteLine("<body>");
Console.WriteLine("<h1>Вас вітає проста CGI-програма!</h1>");
Console.WriteLine("</body>");
Console.WriteLine("</html>");
```

Запустити на виконання в режимі відладки, переконайтеся, що у вихідний потік виводиться правильна директива сервера, після неї порожній рядок і далі синтаксично правильний HTML-документ.

До пункту 3.

Скопіювати створений виконуваний файл в папку, пов'язану з віртуальним каталогом сервера і випробувати роботу, задавши в браузері адресу ресурсу.

При використанні на лабораторній роботі сервера, що знаходиться на ВЦ 418, матеріали розміщувати у віртуальному каталозі WEB_4_7 в папці з вашим прізвищем. Створений в проекті виконуваний файл по локальній мережі помістити в папку на сервері, переконатися в працездатності ресурсу, задавши його адресу в браузері: http://172.16.178.20/WEB_4_7/ім'я_папки/ім'я_ресурсу.

До пункту 4.

При відправці даних з форми методом GET (використовується за замовчуванням) вони поміщаються сервером в змінну оточення QUERY_STRING у вигляді name1=value1&name2=value2& .

Доступ до змінної і перетворення даних в масив рядків (без символів "=" і "&") для подальшого використання може виконуватися так:

```
string  query  = System .Environment.GetEnvironmentVariable("QUERY_STRING ");
char[ ] SplitChars  = new  char[ ] { '=' , '&'  };
string[ ] result  = query .Split(SplitChars );
```

Література: основна [1].

Тема 7. Технології активних сторінок

Лабораторна робота 12. Дослідження взаємодії браузера з сервером при генерації динамічних сторінок ASP

Мета роботи – вивчення можливостей створення динамічних сторінок на сервері з використанням технології ASP і придбання практичних навиків програмування на сервері.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

обґрунтовувати вибір потрібної технології створення WEB-додатків;

створювати динамічні сторінки і обробляти дані з форм, використовуючи засоби програмування на стороні сервера WWW.

Указані вміння надають можливість вирішення таких ***завдань***:

налаштовувати WEB-сервери для роботи зі сторінками, які формуються динамічно;

аналізувати та створювати сайти з використанням технологій активних сторінок ASP;

обробляти данні з форм за допомогою серверних Java-скриптів.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати тему своєї розробки, підготувати необхідні алгоритми і програми.

При виконанні лабораторної роботи:

1. Створити і випробувати роботу простої ASP-сторінки.

2. Розробити варіант ASP-сторінки відповідно до індивідуального завдання (теж саме, що у попередній роботі).

3. Налагодити і випробувати роботу створеної динамічної сторінки у складі сайта на сервері.

Звіт з лабораторної роботи надається у вигляді продукту, який створено при виконанні роботи за кожним пунктом завдання та оформлено за особистим задумом.

Контрольні запитання:

1. Який з підходів до створення динамічних сторінок представляється вам найбільш універсальним? Чому?

2. Який з підходів до створення динамічних сторінок представляється вам найбільш простим в реалізації? Чому?

3. Поясніть зв'язок між розширеннями сервера і технологіями, заснованими на скриптах, вбудованих в сторінку.

4. Дати порівняльну характеристику відомих вам технологій, заснованих на серверних скриптах.

5. Які з серверних скриптів найбільш популярні? Чому?

6. Сформулюйте відмінності у використанні скриптів на стороні сервера і клієнта.

7. Порівняєте можливості, що забезпечуються технологією ASP і CGI.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Простійний серверний скрипт ASP, який забезпечує видачу клієнтові дати:

```
<%@ Language=JavaScript  %>
<HTML >
<HEAD>
<TITLE> Приклад сторінки</TITLE>
</HEAD>
  <BODY >
  <div>Сторінка відправлена    <%= Date () %>
 </div>
  </BODY >
```

До пункту 2.

На ASP-сторінках забезпечується доступ до вбудованих об'єктів, що полегшують отримання відомостей, що відправляються в запиті клієнта, створення відповіді та зберігання відомостей про конкретного користувача, наприклад його налаштування.

Такими об'єктами є таке.

Application – слугує для надання сумісного доступу до інформації всім користувачам даного застосування.

Session – слугує для зберігання відомостей про сеанс конкретного користувача. Змінні, що зберігаються в об'єкті Session, не знищуються під час переходу користувача з однієї сторінки додатка на іншу; значення цих змінних зберігаються протягом всього часу роботи користувача з додатком. Методи об'єкта Session можна також використовувати для примусового завершення сеансу і для налаштування періоду очікування сеансу.

Request – призначений для отримання інформації, переданої в HTTP-запиті. Сюди входять параметри, передані з HTML-форм за допомогою методу POST або GET, модулі налаштування клієнта і клієнтські сертифікати.

Response – використовується для управління інформацією, що відправляється користувачеві. Сюди входить безпосередня відправка даних оглядачеві, перенаправлення оглядача на іншу URL-адресу або установка значень в модулі налаштування клієнта.

Server – надає доступ до методів і властивостей на сервері. Найчастіше використовується метод для створення екземпляра COM-компонента (Server.CreateObject). Інші методи переводять рядки у формат URL або HTML, перетворять віртуальні шляхи у фізичні і встановлюють періоди очікування для сценаріїв.

ObjectContext – використовується для завершення або припинення транзакції, ініційованої сценарієм ASP.

ASPError – слугує для перехоплення помилок ASP і видачі користувачам більш інформативних описів помилок, що виникли.

Детальну інформацію про властивості та методи цих об'єктів можна отримати в основній та додатковій літературі, або в мережі Інтернет.

До пункту 3.

Для перевірки працездатності і відладки створені ASP-сторінці помістити в папку, пов'язану з віртуальним каталогом сервера (ту ж, що використовувалася на попередній лабораторній роботі).

Переконатися в працездатності ресурсу можна задавши його ім'я в браузері.

Література: основна [1].

Лабораторна робота 13. Дослідження взаємодії браузера з сервером при відображенні динамічних сторінок, створених з використання ASP.NET

Мета роботи – вивчення можливостей створення динамічних сторінок ASP.NET і придбання практичних навиків програмування на сервері.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

обґрунтовувати вибір потрібної технології створення WEB-додатків;

створювати динамічні сторінки і обробляти дані з форм, вико ристовуючи засоби програмування на стороні сервера WWW.

Указані вміння надають можливість вирішення таких ***завдань***:

налаштовувати WEB-сервери для роботи зі сторінками, яки формуються динамічно;

аналізувати та створювати сайти з використанням технологій активних сторінок ASP.NET;

виконувати обробку подій клієнта на сервері.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати тему своєї розробки, підготувати необхідні алгоритми і програми.

При виконанні лабораторної роботи:

1. Ознайомитися з процесом створення і використання сторінок ASP .NET.

1.1. Уточнити налаштування IIS для роботи з ASP .NET.

1.2. Створити в середовищі середі VS простий додаток ASP .NET для видачі клієнтові WEB-сторінки.

1.3. Випробувати роботу створеного застосування.

2. Створення ASP .NET додатки за індивідуальним завданням (теж, що і у попередній роботі).

2.1. Використання серверних елементів, що управляють.

2.2. Розміщення програмних кодів в тексті сторінки і в окремому файлі.

2.3. Включення до складу додатку готових HTML-сторінок.

3. Обробка подій.

3.1. Створити сторінку з обробкою подій на сервері.

3.2. Створити сторінку з обробкою подій на клієнтові.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Перерахуєте основні принципи побудови архітектури ASP.NET.

2. Назвіть основні класи компонентів, які використовуються на формах ASP.NET.

3. Перерахуєте основні компонентів, які використовуються на формах ASP.NET.

4. Як здійснюється обробка подій клієнта на сервері?

5. Що таке змінні класу?

6. Порівняєте різні технології створення динамічних сторінок.

7. Що таке AJAX ? Які недоліки ASP.NET дозволяє подолати цей підхід?

Довідкові матеріали до лабораторної роботи

До пункту 1.

Для тестування динамічних сторінок на комп'ютері має бути встановлений WEB-сервер. При використанні Internet Information Services(IIS) для відображення ASP.NET-сторінок необхідно у властивостях віртуальної папки на вкладці **ASP.NET** задати всі поля, включаючи версію.

Робота зі створення WEB-вузла у середовищі Visual Studio достатньо детально розглянута у рекомендованій літературі [5].

До пункту 2.

Код, який виконує сервер, може бути розміщенний як у тексті сторінки, так і в окремому файлі.

Приклад сторінки з кодом, включеним у текст:

```
<%@ Page Language="C#" %>
<html>
<script runat="server">Void Button1_Click(object sender, System.EventArgs e) {
Label1.Text = ("Welcome " + TextBox1.Text);}</script>
<head runat="server">
<title>Basic ASP.NET WEB Page</title>
</head>
<body>
<form id="form1" runat="server">
<h1>Welcome to ASP.NET</h1>
<p>Type your name and click the button.</p>
<p>
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>  <Asp:Button
ID="Button1" runat="server"   Text="Click" OnClick="Button1_Click" />
</p>
<p>
<asp:Label ID="Label1" runat="server"></asp:Label>
</p>
</form>
</body>
</html>
```

Приклад сторінки з кодом в окремому файлі:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>Untitled Page</title>
</head>
<body>
<form id="form1" runat="server">
<div>
<asp:Button ID="Button1" runat="server" Text="Button" onclick="Button1_Click" />
</div>
<asp:Label ID="Label1" runat="server" Text="Нажмите кнопку"></asp:Label>
</form>
</body>
</html>
```

Файл з кодом обробника (default.aspx.cs):

```
using System;
public partial class _Default : System.WEB.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
{

}
protected void Button1_Click(object sender, EventArgs e)
{
Label1.Text = "Ви натиснули ASP-кнопку";
}
}
```

До пункту 3.

Створення обробників подій на сторінці ASP.NET може бути виконане практично так само, як і у традиційній програмі з формою. Однак необхідно знати про деякі особливості обробки подій на сторінках ASP.NET.

Серверні елементи управління містять обмежений набір подій, які зазвичай є подіями натиснення. Деякі серверні елементи управління підтримують події змін. Наприклад, серверний елемент управління CheckBox створює подія CheckedChanged, якщо користувач знімає або зазначає прапорець. Події, які відбуваються часто (і можуть бути викликані без повідомлення про це користувача), наприклад, подія onmouseover, не підтримуються для серверних елементів управління.

Елементи управління і сама сторінка також створюють події життєвого циклу на кожному кроці обробки, Init, Load, PreRender та інші. Наприклад, в події Load сторінки можна задати значення за замовчуванням для елементів управління.

Більш детально про обробку подій можна дізнатися з основної та додаткової літератури, або у мережі Інтернет.

Література: основна [1].

Лабораторна робота 14. Дослідження взаємодії браузера з сервером при відображенні динамічних сторінок, створених з використання мови PHP

Мета роботи – вивчення можливостей створення динамічних сторінок PHP і придбання практичних навиків програмування на сервері.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

обґрунтовувати вибір потрібної технології створення WEB-додатків;

створювати динамічні сторінки і обробляти дані з форм, використовуючи засоби програмування на стороні сервера WWW.

Указані вміння надають можливість вирішення таких ***завдань***:

налаштовувати WEB-сервери для роботи зі сторінками, які формуються динамічно;

аналізувати та створювати сайти за технологією активних сторінок з використанням мови PHP.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Продумати тему своєї розробки, підготувати необхідні алгоритми і програми.

При виконанні лабораторної роботи:

1. Вивчити процес налаштування сервера для роботи з PHP-сторінками.

1.1. Підключити модуль PHP, переконатися у його працездатності.

1.2. Встановити PhpMyAdmin для роботи з БД.

1.3. Встановити MySQL-сервер , випробувати функціонування.

2. Створення PHP-додаток за індивідуальним завданням (теж, що і у попередній роботі).

2.1. Створити додаток застосування в одному файлі.

2.2. Розмістити частини додатка в різних файлах.

3. Робота з базами даних.

3.1. Створити БД (таблицю в тій базі, що існує) для зберігання оброблюваної інформації (використовувати PhpMyAdmin).

3.2. Адаптувати свій додаток для роботи з БД.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Сформулюйте переваги мови PHP для створення динамічних сторінок.

2. Сформулюйте правила розміщення PHP-скриптів на сторінках.

3. Сформулюйте відмінності в синтаксисі PHP і знайомих вам мов C і JavaScript .

4. Назвіть способи здобуття доступу до даних з форм в PHP.

5. Перерахуєте послідовність дій для здобуття доступу до даних в БД. Назвіть функції для підтримки цих дій.

6. Дайте характеристику середовищ для розробки додатків на PHP.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Процес налаштування сервера для роботи з PHP-сторінками полягає у підключенні та налаштуванні модуля PHP. Модуль PHP – це програма, що взаємодіє з WEB-сервером і виконує інтерпретацію коду PHP. Взаємодія (часто говорять підключення) модуля може здійснюватися по різних інтерфейсах, підтримуваним даним сервером. Для IIS це CGI, FASTCGI або ISAPI. Слід зазначити, що FASTCGI розроблений спеціально для IIS з метою подолання недоліків CGI, тому для установки можна сміливо вибирати саме цей варіант.

Установка може проводитися або вручну, шляхом розміщення файлів модуля у відповідних папках на сервері і налаштування конфігураційних файлів, або з використанням програми-інсталятора. Останній варіант простіший, після запуску інсталятора запускається майстер, за допомогою якого вибирається місце розміщення файлів інтерфейс підключення і інші установки, більшість яких можна залишити без змін.

Для перевірки роботи модуля PHP після установки необхідно помістити в кореневу папку WEB-сайта (c:\inetpub\wwwroot) файл test .php такого змісту :

```
<html >
<head >
<title>Test  PHP</title>
</head >
<body >
<?php  phpinfo ()?>
</body >
</html >
```

Якщо все працює правильно, то після відкриття сайта (http://localhost/test.php) в браузері відобразиться сторінка з інформацією про встановлену версію PHP.

Після установки може бути потрібно додаткове налаштування, для якого використовується файл php.ini. Незалежно від платформи і WEB-сервера, використовуваного у поєднанні з PHP, файл php.ini містить однаковий набір стандартних параметрів, що дозволяють управляти режимами роботи модуля при виконанні сценарію.

Найчастіше може виникнути необхідність в зміні таких параметрів :

short_open_tag [on off] – можливість використання коротких тегів <?...?>;

asp_tags [on off] – можливість використання тегів в стилі ASP (<% print "This is PHP code."; %>);

precision [integer] – кількість значущих цифр, що відображаються в дійсних числах;

safe_mode [on off] – безпечний режим (користувач не зможе застосувати сценарій PHP для діставання доступу до іншого файла в системі). Параметр safe_mode працює тільки в CGI-версії PHP.

max_execution_time [integer] – максимальна тривалість виконання сценаріїв PHP в секундах (таке обмеження запобігає поглинання цінних системних ресурсів сценаріями, що містять помилки);

error_reporting [1–8] – рівень видачі повідомлень про помилки в PHP (чим вище значення, тим "чутливіше" PHP реагує на помилки, 1 – звичайні помилки, 2 – попередження, 4 – помилки лексичного аналізатора, 8 – зауваження);

display_errors [on off] - управління виведенням інформації про помилки в браузері.

До пункту 2.

Для включення в зміст сторінки фрагментів, які зберігаються в окремих файлах, в PHP використовуються спеціальні функції:

include( );

include_once( );

require( );

require_once( ).

При використанні цих функцій файл, що включається, може мати довільне розширення. Крім того, якщо його вміст є PHP-кодом, то воно має бути поміщене всередину дескриптора <?php.?>.

Функція include (file) вставляє вміст файла з указаним ім'ям у сценарій при виконанні звернення до неї.

Функція include_once(file) виконує те ж, що і include(), але заздалегідь перевіряє, чи не був він включений раніше. Якщо файл вже був включений, виклик функції ігнорується.

Функція require(file) вставляє вміст у місці виклику, але вставка проводиться до виконання коду (отже, якщо помістити виклик в умовному операторі, код буде вставлений незалежно від результату перевірки умови). Require_once(file) – вставка проводиться тільки один раз.

До пункту 3.

Мова PHP орієнтована на роботу з СУБД MYSQL. Робота з БД організовується через SQL-сервер, причому SQL-сервер у будь-якому випадку розглядається як видалений, тобто для його використання створюється мережеве з'єднання. Таким чином, навіть для автономної відладки на комп'ютері окрім WEB-сервера має бути встановлений і запущений MySQL-сервер. Само створення бази даних та її таблиць може бути виконане як з скрипта, так і з будь-якої програми-менеджера (наприклад, phpMyAdmin).

Якщо база даних вже створена, то в загальному випадку порядок виконання дій такий. Після встановлення з'єднання з сервером, вибирається база даних для роботи. Потім формуються запити на обробку. Для виконання запиту створюється об'єкт, в якому зберігається результат виконання запиту або дані для запису.

Мова PHP містить багато функцій для роботи з БД. Нижче приведені прототипи функцій, що забезпечують приведену послідовність дій.

resource mysql_connect ([string server [, string username [, string password]]]) – встановлює з'єднання з сервером.

bool mysql_close ([resource link_identifier]) – розриває з'єднання з сервером.

bool mysql_select_db (string database_name [, resource link_identifier]) – вибір бази даних для роботи.

resource mysql_query (string query) – відправка запиту сервера (сам запит представляє рядок, складений за правилами SQL).

array mysql_fetch_array (resource result) – розміщення значень полів в асоціативному масиві.

Література: основна [1]; додаткова [5].

Змістовний модуль 4. XML і його використання

Тема 8. Мова розмітки XML

Лабораторна робота 15. Створення та відображення XML-документів

Мета роботи – вивчення синтаксису мови XML, способів відображення XML-документів у браузерах і придбання практичних навиків в роботі з XML-документами.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

аналізувати та використовувати XML-документи при створенні WEB-ресурсів.

Указані вміння надають можливість вирішення таких ***завдань***:

створення XML-документів різного змісту;

відображати XML-документи у браузері за допомогою таблиць стилів;

відображати XML-документи шляхом зв’язування з HTML-сторінкою.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Розробити структуру документа для використання в ході роботи, підготувати декілька варіантів різної складності, підготувати ескіз документа, що ,буде відображатися браузером.

3. Підготувати HTML-сторінку для зв'язування з XML-документом.

При виконанні лабораторної роботи:

1. Створити XML-документ (для створення використовувати блокнот або XML-редактор).

1.1. Створити XML-документ з іменами тегів кирилицею.

1.2. Створити XML-документ з іменами тегів латиницею.

1.3. Порівняти відображення документа в різних браузерах.

2. Створити таблиці стилів для відображення документа відповідно до власного задуму.

2.1. Створити таблицю для кириличних тегів.

2.2. Створити таблицю тегів латиницею.

2.3. Порівняти результати відображення в різних браузерах.

3. Пов'язати для відображення XML-документ з HTML-сторінкою з використанням об'єкта DSO.

3.1. Встановити зв'язок записів з таблицею для відображення всього документа і груп записів.

3.2. Встановити зв'язок між окремими елементами для відображення записів, зображень, посилань.

3.3. Порівняти результати відображення в різних браузерах.

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Яку мету ставили перед собою розробники XML?

2. Сформулюйте основні правила розмітки з використанням XML.

3. Для чого слугує секція <!CDATA[....]]>?

4. Які частини можна виділити в структурі XML-документа?

5. Назвіть області, де може знайти застосування мова XML.

6. Які дії виконує браузер при відображенні XML-документів?

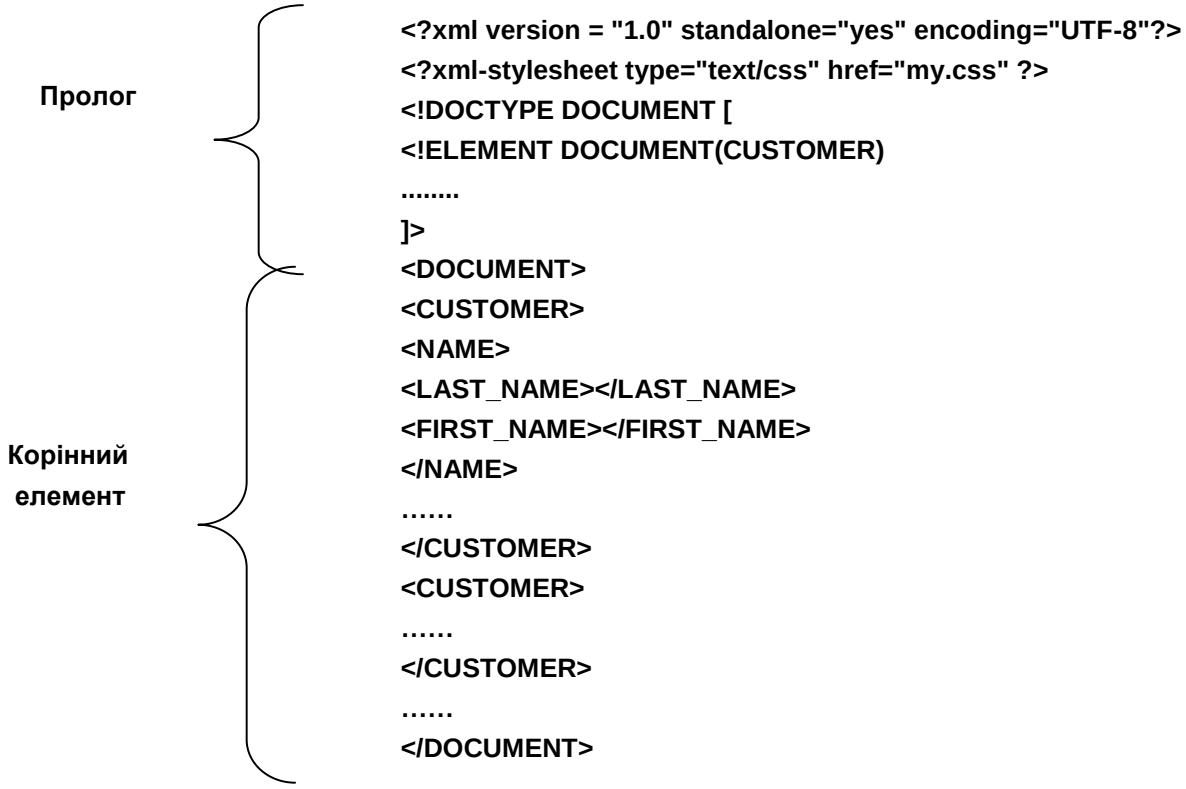
7. За наслідками виконання роботи сформулюйте свої рекомендації з використання таблиці стилів спільно з XML-документами.

8. Які обмеження накладаються на XML-документ при використанні зв'язування? Наскільки універсальний цей метод відображення?

Довідкові матеріали до лабораторної роботи

До пункту 1.

У загальному вигляді XML-документ має таку структуру:



До пункту2.

Для управління відображенням XML-документа в браузері використовуються ті ж властивості, що й для HTML. Елементи, що створюються тегами, можна представляти у вигляді списків, позиціонувати, створювати фон, змінювати шрифт, керувати текстом та виконувати інше форматування.

До пункту3.

Для використання зв’язування XML-документа з HTML-сторінкою необхідно включити в сторінку HTML-елемент з ім'ям XML.

```
<HTML>
<HEAD>
<TITLE>Book Description</TITLE>
</HEAD>
<BODY>
<XML ID="dsoBook" SRC="Book.xml"></XML>
<!-- другие элементы HTML … -->
</BODY>
</HTML>
```

Коли Internet Explorer відкриває HTML-сторінку, його вбудований XML-процесор синтаксично аналізує XML-документ, перевіряє, чи є документ коректно сформованим (при виявленні помилки відображення зупиняється), і створює програмний об'єкт, який має назву DSO (Data Source Object, об'єкт вихідних даних). Об'єкт зберігає дані XML і забезпечує доступ до них. Дані зберігаються як набір записів і їх полів.

DSO дозволяє безпосередньо здійснювати доступ і маніпулювання наявним набором записів за допомогою ряду методів (таблиця 21), властивостей (табл. 22) і подій.

Таблиця 21

Методи об'єкта TABLE

Метод елемента TABLE	Эфект	Приклад виклику
FirstPage	Відображає першу сторінку записів	Table1.firstPage()
previousPage	Відображає попередню сторінку записів	Table1.previousPage()
NextPage	Відображає наступну сторінку записів	Table1.nextPage()
LastPage	Відображає останню сторінку записів	Table1.lastPage()

Таблиця 22

Властивості для зв'язування елементів

HTML-елемент	Властивість для зв'язування	Облік розмітки HTML	Оновлення
A	href	Ні	Ні
DIV	innerHTML та innerText	Так	Ні
IMG	src	Ні	Ні
SPAN	innerText та innerHTML	Так	Ні
TEXTAREA	value	Ні	Так

Подіями є певні зміни стану (наприклад, зміна значень запису), якими можна управляти за допомогою функції сценарію, створеного для сторінки.

Наприклад, використання однієї HTML-таблиці для відображення простого набору записів:

```
<HTML>
<HEAD>
<TITLE>Book Inventory</TITLE>
</HEAD>
```

```
<BODY>
<XML ID="dso1" SRC="pr1ru.xml"></XML>
<H2>Список Литературы</H2>
<TABLE DATASRC="#dso1" BORDER="1" CELLPADDING="5">
<THEAD>
<TH>Автор</TH>
<TH>Название</TH>
<TH>издательство</TH>
<TH>год издания</TH>
<TH>кол. страниц</TH>
</THEAD>
<TR ALIGN="left">
<TD><SPAN DATAFLD="автор"  STYLE="font-style:italic"></SPAN></TD>
<TD><SPAN DATAFLD="название"></SPAN></TD>
<TD><SPAN DATAFLD="издательство"></SPAN></TD>
<TD><SPAN DATAFLD="год"></SPAN></TD>
<TD><SPAN DATAFLD="кол_страниц"></SPAN></TD>
</TR>
</TABLE>
</BODY>
```

Якщо XML-документ містить багато записів, то можна використовувати посторінкове виведення групи записів за один раз замість відображення всіх записів одночасно. Для управління посторінковим відображенням необхідно в тег TABLE додати атрибут DATAPAGESIZE="кол_зап" і використовувати методи об'єкта TABLE, фактично включивши в сторінку скрипти.

Література: основна [1]; додаткова [6].

Тема 9. Використання XHTML

Лабораторна робота 16. Дослідження особливостей XHTML-документів

Мета роботи – дослідження особливостей XHTML-документів і придбання практичних навиків в роботі з ними.

Дане лабораторне заняття забезпечує напрацювання таких ***умінь***:

обирати засоби, методи і технології для створення WEB-сторінок і WEB-сайтів;

обґрунтовувати вибір між мовами HTML та XHTML.

Указані вміння надають можливість вирішення таких ***завдань***:

обґрунтування вибору основних рішень зі створенню WEB-сайтів;

використання переваг мови XHTML.

Завдання для лабораторної роботи

При підготовці до лабораторної роботи:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Розробити структуру документів для використання в ході роботи, підготувати декілька варіантів різної складності.

3. Підготувати DTD-визначення .

При виконанні лабораторної роботи:

1. Створення XHTML-документа.

1.1. Створити документ.

1.2. Перевірити валідність.

1.3. Перевірити відображення в різних браузерах, порівняти результати

2. Розширення XHTML

2.1. Додати в документ власний тег і значення його властивостей в таблицю стилів, перевірити відображення в різних браузерах.

2.2. Доповнити визначення DTD, перевірити валідність.

2.3. Перевірити розпізнавання документів браузерами при різних заголовках (вид специфікації і режим сумісності).

Звіт з лабораторної роботи представляється у вигляді сторінок, які створені при виконанні кожного пункту за особистим замислом та оформленням.

Контрольні запитання:

1. Які переваги надає розробникам XHTML?

2. Сформулюйте основні правила розмітки з використанням XHTML

3. Сформулюйте основні відмінності XHTML і HTML.

4. З якої версії в XHTML включена підтримка модульності? У чому вона полягає?

5. Як включити новий елемент розмітки в XHTML?

6. Які ознаки використовуються браузерами для розпізнавання XHTML-документа?

7. Сформулюйте свою власну думку стосовно сфери використання XHTML.

Довідкові матеріали до лабораторної роботи

До пункту 1.

Приклад прологу XHTML-документа:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN
"http://www.w3.org/TR/xhtml1-transitional.dtd">
```

Приклад корінного елемента XHTML-документа:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
…
</html>
```

До пункту 2.

Приклад файла DTD з розширенням визначень:

```
<!ELEMENT underl_redt (#PCDATA)>
<!ATTLIST underl_red text_attrib CDATA #IMPLIED >
<!ENTITY % XHTMLTransDTD PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://w3.v3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
%XHTMLTransDTD;
```

До пункту 3.

Варіанти прологу для XHTML-документів:

1. XHTML 1.0 Transitional doctype сумісно з XML-оголошенням

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

2. XHTML 1.0 Transitional doctype без XML-оголошення

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

3. **<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"**
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

4. **<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"**
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

5. HTML 4.0 Transitional doctype з URL

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

6. HTML 4.01 Transitional doctype з URL

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/1999/REC-html401-19991224/loose.dtd">
```

7. HTML 4.01 Transitional doctype з URL

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

8. HTML 4.01 Transitional doctype без URL

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

Література: основна [1]; додаткова [6].

Рекомендована література

Основна

1. Молчанов В. П. Технології WEB-дизайну : конспект лекцій / В. П. Молчанов. – Х. : Вид. ХНЕУ, 2011. – 212 с.

Додаткова

2. Дронов В. Javascript в WEB-дизайне / В. Д. Дронов. – СПб. : БХВ-Петербург, 2001. – 880 с.

3. Методичні рекомендації по виконанню лабораторних робіт з навчальної дисципліни "Основи проектування WEB-видань" для студентів спеціалізації "Комп'ютеризовані технології та системи видавничо-поліграфічних виробництв" усіх форм навчання / укл. В. П. Молчанов, Т. Ю. Андрющенко. – Х. : Вид. ХНЕУ, 2009. – 84 с.

4. Молчанов В. П. Основи проектування WEB-видань : конспект лекцій / В. П. Молчанов. – Х. : Вид. ХНЕУ, 2008. – 168 с.

5. Томсон Л. Разработка WEB-приложений на PHP и MySQL / Л. Томсон, Л. Веллингтон ; пер. с англ. – 2-е изд., испр. – СПб. : ООО "ДиаСофтЮП", 2003. – 672 с.

6. Холзнер С. XML : Энциклопедия / С. Холзнер. – 2-е изд. – СПб. : Питер, 2004. – 1101 с.

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни "ТЕХНОЛОГІЇ WEB-ДИЗАЙНУ"

для студентів напряму підготовки 6.051501 "Видавничо-поліграфічна справа" спеціалізації "Технології електронних мультимедійних видань" усіх форм навчання

Укладач **Молчанов** Віктор Петрович

Відповідальний за випуск **Пушкар О. І.**

Редактор **Бутенко В. О.**

Коректор **Бриль В. О.**

План 2012 р. Поз. № 376.

Підп. до друку Формат 60×90 1/16. Папір MultiCopy. Друк Riso.

Ум.-друк. арк. 4,0. Обл.-вид. арк. 5,0. Тираж прим. Зам. №

Видавець і виготівник – видавництво ХНЕУ, 61166, м. Харків, пр. Леніна, 9а

Свідоцтво про внесення до Державного реєстру суб'єктів видавничої справи ***Дк № 481 від 13.06.2001 р.***