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

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Методичні рекомендації
до виконання лабораторних робіт
з навчальної дисципліни
"ЗАСОБИ СИСТЕМ ОБРОБКИ ДАНИХ"

**для студентів галузі знань 0515
"Видавничо-поліграфічна справа"
всіх форм навчання**

Харків. Вид. ХНЕУ, 2013

Затверджено на засіданні кафедри комп'ютерних систем і технологій.
Протокол № 4 від 04.12.2012 р.

Укладач Молчанов В. П.

М54 Методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни "Засоби систем обробки даних" для студентів галузі знань 0515 "Видавничо-поліграфічна справа" всіх форм навчання / укл. В. П. Молчанов. – Х. : Вид. ХНЕУ, 2013. – 40 с. (Укр. мов.)

Подано методичні рекомендації до лабораторних робіт зі створення систем обробки даних із використанням мови XML, які містять завдання та основний довідковий матеріал, необхідний для їх виконання.

Рекомендовано для студентів галузі знань 0515 "Видавничо-поліграфічна справа" всіх форм навчання.

Вступ

Вивчення навчальної дисципліни "Засоби систем обробки даних" має на меті формування компетентностей, пов'язаних з використанням мови XML, її додатків та інших взаємопов'язаних технологій, відповідно до сьомого рівня Національної рамки кваліфікацій. Під час навчання студенти повинні набути такі вміння:

обґрунтовувати склад і тип компонентів системи;

вибирати формат документів для обміну між додатками обробки;

відображати XML-документи найбільш підходящим методом;

розробляти DTD для XML-документів та виконувати перевірку валідності і коректності за допомогою різних засобів;

створювати XSLT для перетворень і обробки XML-документів;

перетворювати HTML-документи на XHTML;

створювати найпростіші XSL-FO документи;

перетворювати XSL-FO документи в інші формати.

З метою їх набуття до складу навчальної дисципліни включені лабораторні роботи, спрямовані на вирішення відповідних типових задач. Для кожної роботи в методичних рекомендаціях сформульовано перелік таких задач, визначений порядок дій і наведений 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 даними і тому подібне.

Основною формою звіту у більшості робіт є створені підсумкові документи і проміжні файли з даними. Під час захисту 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. Засоби системи обробки даних

Тема 1. Системи обробки даних

Лабораторна робота 1. Засоби систем обробки даних

Мета: Ознайомлення з архітектурними особливостями СОД, налаштування і аналіз функціонування компонент.

Лабораторне заняття забезпечує напрацювання таких ***умінь***:

обґрунтовувати склад і тип компонентів системи;

здійснювати аналіз та налагодження компонент систем обробки даних.

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

вибір найбільш придатних засобів для створення СОД;

встановлення та налаштування компонент СОД;

організація спільного функціонування компонент;

оцінка ефективності прийнятих рішень.

Література: [1; 3 – 5].

Завдання лабораторної роботи

При підготовці до лабораторної роботи необхідно:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Ознайомитися з теоретичними матеріалами, що стосується найбільш популярних WWW і SQL серверів.

3. Підготувати інсталяції необхідних засобів, дані для перевірки їх функціонування.

При виконанні лабораторної роботи:

1. Встановлення та налаштування WWW сервера.

1.1. Запустити віртуальну машину (наприклад, VMware Workstation) зі встановленою в ній ОС Windows, при необхідності інсталювати ОС.

1.2. Провести інсталяцію сервера (Internet Information Services або Apache) у віртуальному середовищі Windows.

1.3. Налаштувати сервер на доступ до створеної HTML сторінки або сайта. До IIS підключити PHP, переконайтеся в його працездатності.

1.4. Вивчити різні налаштування і режими доступу до сервера.

2. Встановлення та налаштування SQL сервера.

2.1. Провести інсталяцію SQL сервера (Microsoft SQL 2005 Server Express або MySQL).

2.2. Встановити програму-менеджер (Microsoft SQL Server Management Studio Express 2005 або PhpMyAdmin), переконатися в працездатності.

2.3. Виконати різні дії з БД (підключення, редагування тощо), підготувати базу даних для наступного пункту завдання.

3. Організація спільної роботи серверів.

3.1. Створити активну сторінку для забезпечення доступу до створеної БД.

3.2. Виконати налаштування серверів для забезпечення спільної роботи.

Звіт з лабораторної роботи подається у вигляді встановлених і налаштованих засобів, а також письмових висновків з кожного пункту.

Контрольні запитання

1. Тонкий і товстий клієнти. Що це таке і яка між ними різниця?

2. Що таке Microsoft SQL Server і Microsoft SQL Server Express, яка між ними різниця? У яких організаціях і для виконання яких завдань доцільно використовувати SQL Server і SQL Server Express?

3. Сформулюйте рекомендації щодо областей застосування SQL Server Express і MySQL.

4. Що таке Microsoft SQL Server Management Studio Express і навіщо вона потрібна?

5. Чим відрізняється стиск БД від стиснення файла БД?

6. Для чого виконується резервне копіювання БД? Назвіть його основні параметри. Які існують методи створення резервної копії і в чому їх різниця?

7. Коротко опішить процедуру інсталяції та базової налаштування обраного SQL сервера.

8. Що таке Internet Information Services?

9. Сформулюйте рекомендації щодо областей застосування IIS та Apache.

10. Що таке консоль управління Microsoft Management Console? У яких випадках її доцільно використовувати?

11. Коротко опішить процедуру інсталяції та базового налаштування обраного WWW сервера.

12. Для чого виконується резервне копіювання БД? Назвіть його основні параметри. Які існують методи створення резервної копії і в чому їх різниця?

Довідкові матеріали до лабораторної роботи

До пункту 1.

IIS (Internet Information Services) – пропріетарний набір серверів для декількох служб Інтернету від компанії Майкрософт, поширюється з операційними системами сімейства Windows NT.

Основним компонентом IIS є WWW сервер (служба WWW), який надає клієнтам доступ до файлів (Web-сторінок) по протоколах HTTP, HTTPS, FTP, POP3, SMTP, NNTP та ін. У цей час можуть зустрітися версії 5, 6 і 7.

Сервер WWW IIS підтримує кілька різних технологій створення Web-додатків:

ASP.NET – це технологія, розроблена Microsoft; для IIS – це основне на сьогоднішній день засіб створення Web-додатків і Web-служб. Функціонування додатків, створених за цією технологією, вимагає установки на комп'ютері програмної платформи. NET Framework (входить у поставку IIS, починаючи з версій 6.0 для інших версій необхідно встановлювати окремо).

ASP – це попередня ASP.NET технологія створення динамічних Web-сторінок на основі сценаріїв. Входить у поставку IIS, починаючи з версії 3.0.

CGI – це стандартний протокол взаємодії Web-додатка з сервером, може слугувати основою для низькорівневої технології динамічної генерації Web-сторінок.

FastCGI – поліпшений протокол взаємодії сервера і додатка, призначений для тих же цілей, що і CGI.

ISAPI – це розширення API сервера для доступу до всіх можливостей IIS, забезпечує можливість розробки Web-додатків, тісно взаємодіє з сервером, перевизначення частини функцій IIS і додавання нових функцій (зазвичай пов'язаних з генерацією контенту). Підсистема виконання скриптів ASP і підсистема ASP.NET виконані як модулі ISAPI.

SSI – це включення в одні сторінки тексту з інших сторінок (технологія вважається застарілою).

Усі інші технології є надбудовами, працюючими через CGI, FastCGI або ISAPI. Зокрема, для використання технології на основі PHP до сервера повинен бути підключений відповідний модуль, що використовує один з цих інтерфейсів.

Установка Web-сервера IIS. Установка сервера в середовищі ОС Windows 7 виконується в такій послідовності (установка в Windows XP відрізняється незначними деталями).

Послідовно обираємо в меню **Пуск** -> **Панель управления** -> **Программы и компоненты** -> **Включение или отключение компонентов Windows**. У відкритому вікні вибираємо розділ Служби IIS, розкриваємо і вибираємо потрібні компоненти (рис. 1).

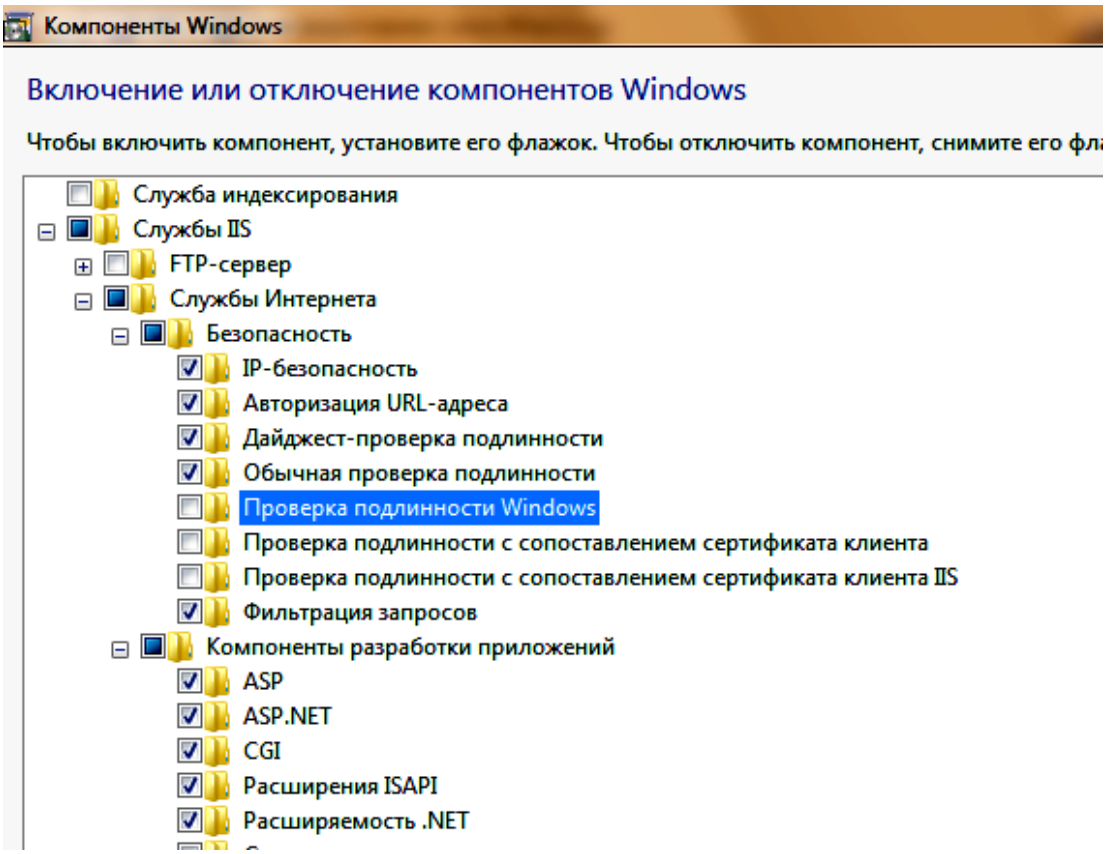


Рис. 1. **Вікно включення компонентів**

Можна використовувати такі установки:

- **Безопасность** – всі компоненти, окрім "Проверка подлинности с сопоставлением сертификата …".
- **Компоненты разработки приложений** – можна включити всі.
- **Общие функции http** – можна включити всі.
- **Проверка работоспособности и диагностика** – "Ведение журнала HTTP" і "Монитор запросов".
- **Функции повышения быстродействия** – можна не включати.
- **Средства управления веб-сайтом** – включити тільки "Консоль управления IIS".

Натиснути Ок. Перезавантажити комп'ютер.

Створення Web-сайта. Для додавання сайта на сервер необхідно відкрити Консоль управління IIS (**КМ Компьютер** -> **Управление**), розкрити групу "Службы и приложения", вибрати "Диспетчер служб IIS". У вікні Підключення вибрати папку Сайти, потім у правому вікні Действия – "Добавить веб-сайт". У вікні (рис. 2) задати ім'я сайта і місце розташування його файлів (за замовчуванням c: \ inetpub \ wwwroot).

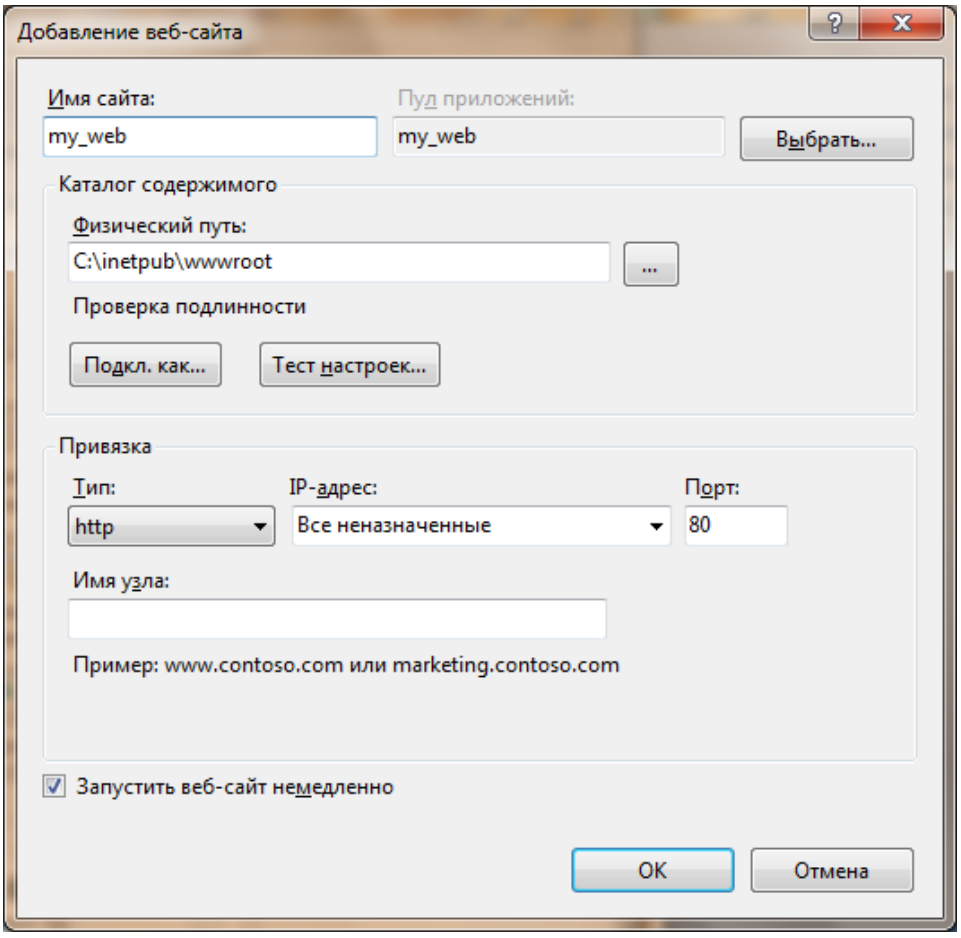


Рис. 2. **Додавання Web-сайта**

Усі сайти розташовуються на **localhost** і відрізняються номером порту (http://localhost:85), можна поміщати фізичні папки сайта всередину wwwroot, тоді доступ буде http://localhost/my_site.

Для перевірки роботи сервера в папку сайта помістити файл index.htm і набрати в браузері http://localhost.

Примітка. *Для того, щоб сайт був доступний з зовні, необхідно відкрити 80-й порт для вхідних з'єднань. Для штатного брандмауера Windows 7:* ***Панель управления -> Система и безопасность -> Брандмауэр Windows -> Дополнительные параметры.*** *У списку знайти і включити правило Служби Інтернету (вхідний трафік HTTP).*

Установка модуля PHP. Скачати необхідний реліз можна на сайті http://windows.php.net/download/. Запустити установник, у першому вікні майстра вибирається місце для файлів (наприклад, Program File / PHP), в наступному вікні – інтерфейс для підключення модуля, рекомендується вибрати IISFastCGI (рис. 3).

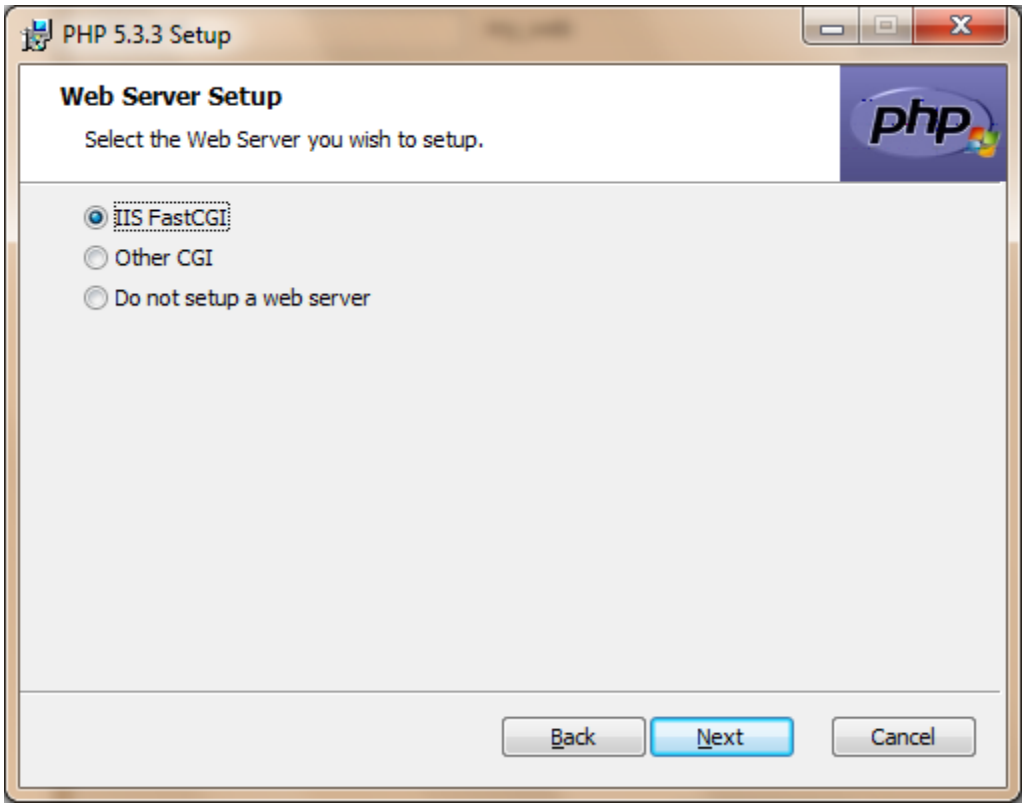


Рис. 3. **Вікно вибору підключення**

В інших вікнах можна залишити значення за замовчуванням.

Примітка. *Для 64-бітної Windows необхідно дозволити виконання 32-бітних додатків (**Пулы приложений** -> **DefaultAppPool** -> **Дополнительные параметры** -> **Общие** -> **Разрешить выполнение 32-битных приложений** -> **True**).*

Для перевірки роботи модуля PHP необхідно помістити в кореневу папку Web-сайта (c: \ inetpub \ wwwroot) файл index.php з таким змістом:

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

Якщо все працює правильно, то після відкриття сайта в браузері (http://localhost/index.php) відобразиться сторінка з інформацією про установку PHP (рис. 4).

Рис. 4. **Інформація про установку PHP**

До пункту 2.

При виборі мережевої СУБД, орієнтованої на створення невеликих за обсягом баз даних (одиниці Гбайт), існує два вільно розповсюджуваних приблизно рівноцінних варіанти: MySQL Server і Microsoft SQL Server Express Edition. Перший – досить універсальне рішення, підтримуване співтовариством незалежних розробників і окремими компаніями. Другий краще інтегрується з технологіями і продуктами Microsoft, що цілком природно. Обидва сервера встановлюються і налаштовуються з використанням майстрів.

Встановлення та налаштування MySQL Server. При установці вибирається тип установки (рекомендується Custom), каталог для файлів і автоматичний перехід до налаштування (відповідний перемикач у вікні).

У ході налаштування можна вибрати деталізовану (Detailed Configuration) або стандартну (Standard Configuration) конфігурацію. У ході деталізованої налаштування у відповідних вікнах потрібно вибрати тип сервера (рекомендується Developer Machine), тип використовуваних таблиць (рекомендується Multifunctional Database), місце розташування файлів бази даних, тип і максимальне число підключень. Крім того, рекомендується включити підтримку TCP/IP з'єднань (Enable TCP/IP Networking), вказати порт, через який вони будуть здійснюватися (стандартним для сервера MySQL є 3306), включити режим строгої відповідності стандарту SQL (Enable Strict Mode). В окремих вікнах вибирається кодування символів (можна вибрати Manual Selected Default Character Set/Collation і cp1251) і режими запуску сервера (включити Install As Windows Service, відключити

Launch the MySQL Server automatically, включити Include Bin Directory in Windows PATH). На останньому кроці встановлюється пароль для користувача root (рис. 5).

Рис. 5. **Вікно задавання пароля для користувача root**

Після успішного виконання всіх налаштувань вікно майстра має набути такого вигляду (рис. 6).

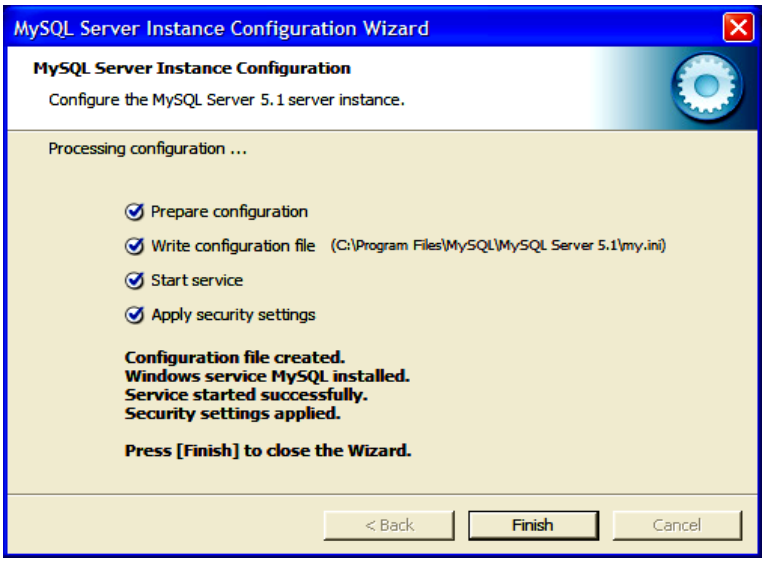


Рис. 6. **Вікно завершення налаштувань**

У цьому вікні рекомендується звернути увагу на рядок Write configuration file, який вказує на місце розташування конфігураційного файла MySQL – my.ini, який при необхідності можна редагувати.

Установлення та налаштування оболонки для маніпулювання базами даних.

Для виконання різних дій з базами даних на MySQL створено достатньо багато різного роду оболонок (наприклад, SQLyog, PhpMyAdmin та ін.)

При орієнтації на технологи PHP доцільно зупинити вибір на PhpMyAdmin, виконаної саме на PHP. Для Microsoft SQL Server Express необхідно встановлювати Microsoft SQL Server Management Studio Express.

Для установки PhpMyAdmin необхідно помістити в папку, відповідну localhost, папку PhpMyAdmin з дистрибутивом. Після цього необхідно створити файл config.inc.php і помістити в нього вміст зразка конфігураційного файла з config.sample.inc. У створеному таким чином конфігураційному файлі коригуються такі директиви:

$ cfg ['Servers'] [$ i] ['auth_type'] = 'config';

$ cfg ['Servers'] [$ i] ['host'] = '127 .0.0.1 ';

Ім'я користувача і пароль повинні відповідати тим, що задавалися при налаштуванні сервера:

$ cfg ['Servers'] [$ i] ['user'] = 'root';

$ cfg ['Servers'] [$ i] ['password'] = '1 ';

Цю директиву помістити в коментар:

/ / $ cfg ['blowfish_secret'] ='';

Відкоригований файл зберігається.

Для перевірки спільної роботи WEB і MySQL серверів з оболонкою PhpMyAdmin необхідно набрати в адресному рядку браузера http://localhost/PhpMyAdmin/. Вид вікна приведений на рис. 7.

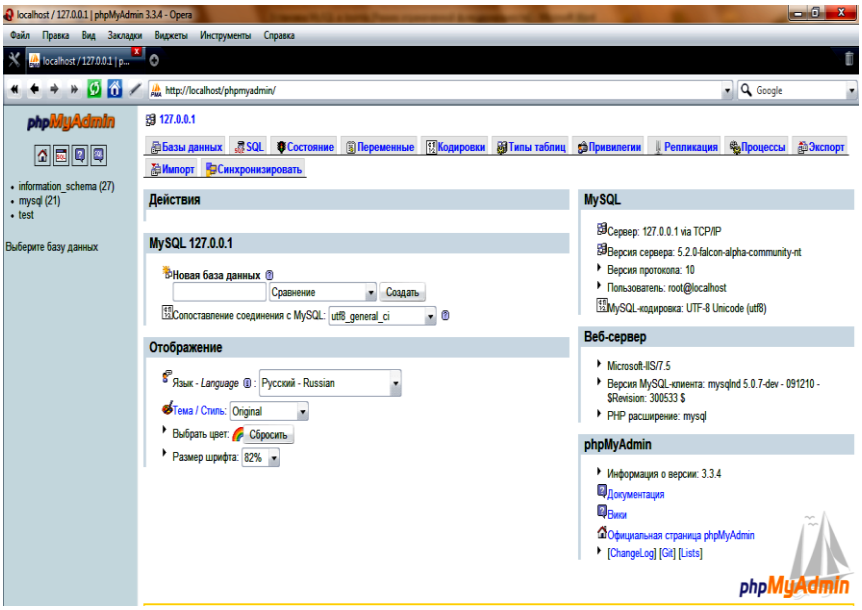


Рис. 7. **Вид вікна оболонки PhpMyAdmin**

До пункту 3.

Правильна робота оболонки PhpMyAdmin вже свідчить про нормальне спільне функціонування серверів IIS і MySQL.

Однак можна використовувати і одну зі спеціально розроблених активних сторінок. Наприклад, сторінку для реєстрації користувача:

```
<html>
<head>
<title>Страница для проверки</title>
</head>
<body>
   <h1>Страница для проверки совместной работы WWW и MySQL серверов</h1>
<p>Если вы зарегистрированный пользователь, выберите режим доступ и введите имя и пароль для доступа</p>
<p>Если нет - выберите режим регистрация и введите имя и пароль для регистрации</p>
<form action='p_test.php' method=POST>
<p><input type="radio" name="mod" value="m1">регистрация </p>
<p><input type="radio" name="mod" value="m2">доступ</p>
<p>имя <input type=text name='name'></p>
<p>пароль <input type=text name='pas'></p>
<input type=submit value='Отправить'>
<input type=reset value='сброс'>
</form>
<?php
 $u_login=$_REQUEST["name"];
 $u_pas=$_REQUEST["pas"];
 $u_mod=$_REQUEST["mod"];
 if (!(($u_login=="")||($u_pas=="")||($u_mod=="")))
    {
   $hostname="127.0.0.1";
   $username="root";
   $password="1";
   $dbName="test";
   mysql_connect($hostname,$username,$password) OR DIE ("Проблемы при создании соединения");
   mysql_select_db($dbName) or die("Проблема при доступе к БД".mysql_error());
```

```
if ($u_mod=="m1"){
    $query = "INSERT INTO sp VALUES('$u_login','$u_pas')";
    $R=mysql_query($query);
    if ($R)echo "<p>Регистрация выполнена</p>";
    else echo "<p>Проблема при записи в БД</p>".msql_error();
            }
if ($u_mod=="m2"){
    $R=@mysql_query("SELECT * FROM sp");
    if (!$R) {echo "Проблема доступа к записям".msql_error();exit();};
    while ($T=mysql_fetch_array($R))
        if (($T[log]==$u_login) && ($T[pas]==$u_pas)){ echo "<p>Доступ разрешен</p>"; exit(); };
    echo '<p>Зарегистрируйтесь</p>';
        }
 }
 else echo "<p>Заполните форму!</p>";
?>
</body>
</html>
```

Перед зверненням до сторінки необхідно за допомогою будь-якого засобу (наприклад, PhpMyAdmin) створити на сервері базу даних **test**, в ній таблицю **sp** з двома полями **log** і **pas** типу **char**.

Тема 2. Робота з XML-документами

Лабораторна робота 2. Створення та відображення XML-документів

Мета: вивчення синтаксису мови XML, засобів створення і способів відображення XML-документів у браузерах.

Лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати документи мовою XML;

відображати XML-документи різними засобами у найбільш поширених браузерах;

використовувати створені у різних середовищах XML-документи.

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

планування робіт щодо створення та використання документів з застосуванням мов розмітки тексту;

створення документів для різних наочних областей;

організація спільного використання документів за допомогою різних засобів.

Література: [1; 5].

Завдання лабораторної роботи

При підготовці до лабораторної роботи необхідно:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Розробити структуру документів для використання в ході роботи, підготувати кілька варіантів різної складності, підготувати ескіз відображуваного браузером документа.

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

При виконанні лабораторної роботи:

1. Створити XML-документ з використанням різних засобів.

1.1. Ознайомитися з можливостями різних засобів для створення XML-документів (блокнот, XML-редактори, середовища розробки).

1.2. Створити XML-документ з іменами тегів кирилицею.

1.3. Створити XML-документ з іменами тегів латиницею.

1.4. Порівняти відображення документів у різних браузерах.

2. Створити таблиці стилів для відображення документів відповідно з власних задумів.

2.1. Створити таблицю для кириличних тегів.

2.2. Створити таблицю тегів латиницею.

2.3. Порівняти результати відображення документів у різних браузерах.

3. Зв'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 подається у вигляді створених документів, а також власних письмових висновків щодо кожного з виконаних пунктів завдання.

Контрольні запитання

1. Які цілі ставили перед собою розробники XML?

2. Сформулюйте основні правила розмітки з використанням XML.

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

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

5. Які атрибути можуть бути присутніми в XML-оголошенні?

6. Як виглядають інструкції щодо обробки XML-документів, для чого вони призначені?

7. Назвіть області, де може знайти застосування мову XML.

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

9. За результатами виконання роботи сформулюйте свої рекомендації щодо використання таблиць стилів спільно з XML-документами.

10. Які обмеження накладаються на XML-документ при використанні зв'язування?

11. На який об'єктної моделі засновано зв'язування? Наскільки універсальний цей метод відображення?

Довідкові матеріали до лабораторної роботи

До пункту 1.

XML-документи можуть створюватися як за допомогою спеціалізованих засобів (XML-редакторів, середовищ розробки), так і звичайних текстових редакторів (Блокнот та ін.) Зрозуміло, ступінь підтримки цього процесу в них різна. XML-редактори, такі, як Altova XMLSpy, XML Copy Editor, XMLPad, окрім зручності представлення і набору тексту, забезпечують ряд додаткових функцій (перевірка коректності та валідності, XSLT трансформації і т. д.). XML-документи можна створювати і в найбільш популярних середовищах розробки Microsoft Visual Studio і Dreamweaver.

У найзагальнішому випадку XML-документ має вигляд, представлений на рис. 8.

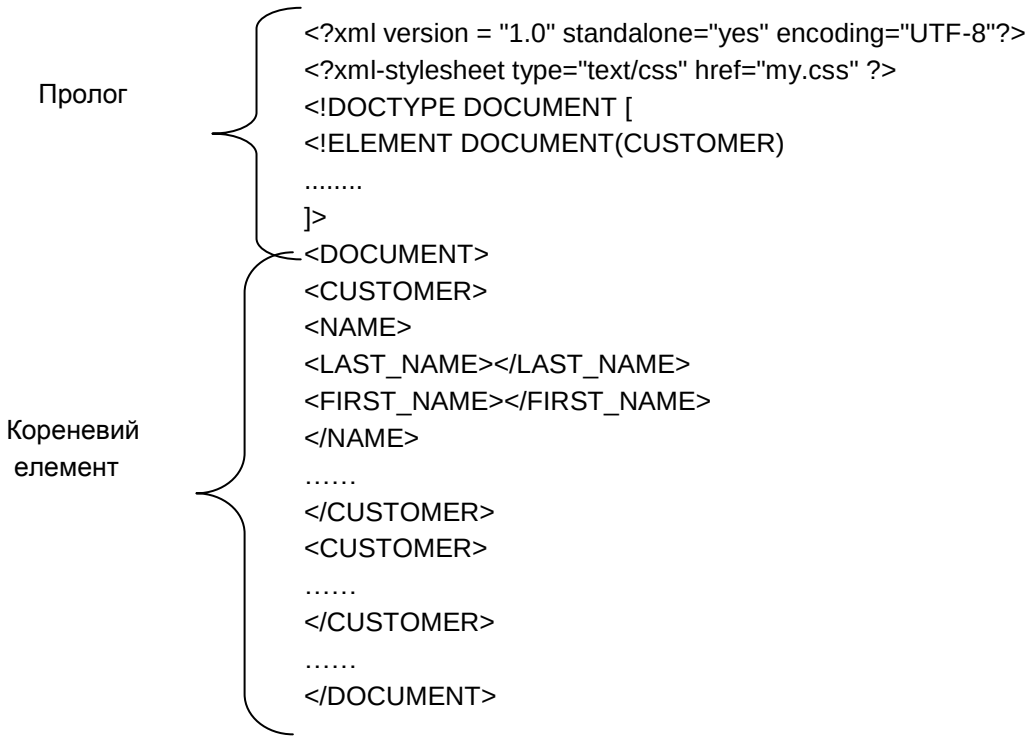


Рис. 8. **XML-документ**

До пункту 2.

Найбільш простим способом описати, як повинні відображатися елементи XML-документа в браузері, є створення таблиці стилів і скріплення її з XML-документом. Сучасні Web-браузери забезпечують високий рівень підтримки каскадних таблиць стилів, крім того, зберігання інструкцій щодо відображення окремо від самого XML-документа підвищує гнучкість використання і полегшує роботу з ним.

Специфікація CSS містить велику кількість різних селекторів. Зокрема, імена тегів (p,h1), класові (.cl) і id-селектори (#id). Проте при роботі з xml-документами класові та id-селектори у такому вигляді не працюють. Для зв'язку правил з елементами більш за все підходить селектор атрибутів. Він конструюється з імені елемент і імені атрибута в квадратних дужках:

p[class] — усі теги p, що мають атрибут class;

p[class="c1"] — усі теги p, що мають атрибут class із значенням c1;

*[id="id1"] — усі теги, що мають атрибут id із значенням id1.

Використовувати можна будь-які атрибути, а не лише class та id.

Далі у табл. 1 – 6 наведені основні властивості елементів.

Таблиця 1

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, 25 mm; % 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

Таблиця 2

Властивості списків

Ім'я	Значення	Примітка
list-style-image	Url ("шлях"), none	Зображення маркера
list-style-position	Outside, inside	Розміщення маркера
list-style-type	Disc, circle, square ...	Вид маркера

Таблиця 3

Властивості для управління фоном

Ім'я	Значення	Примітка
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

Таблиця 4

Властивості, використовувані для позиціонування

Ім'я	Значення	Примітка
Position	Absolute, fixed, relative, static	Задає систему позиціонування
Top, right, left, bottom	<число>	Визначають положення елемента
Float	Right, left, none	Обтікання
Z-index	<число>	Управляє порядком накладення елементів
Visibility	Visible, hidden, collapse	Управління видимістю елементу

Таблиця 5

Властивості шрифтів

Ім'я	Значення	Примітка
Font-family	<список імен>	Список імен шрифтів або сімейств
Font-size	xx-small, x-small, small ... або <число>	Розмір шрифту
Font-style	Normal, italic, oblique	Зображення шрифту
Font-variant	Normal , small-caps	Представлення рядкових літер
Tont-weight	Normal, bold, bolder, lighter, 100, 200 ...	Насиченість шрифту

Таблиця 6

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 щодо батьківського елемента або навколишнього тексту
word-spacing	<число>, normal	Інтервал між словами

До пункту 3.

Метод зв'язування даних запропонований Microsoft і підтримується тільки в її продуктах.

Зв'язування даних працює лише з XML-документом, який симетрично структурований, тобто елементи документа можуть бути інтерпретовані як набір записів і полів. У простому випадку такий документ складається з кореневого елемента, що містить кілька елементів однакового типу (записи),

кожен з яких має однаковий набір дочірніх елементів, усі з яких містять символьні дані (поля).

Щоб відобразити XML-документ на HTML-сторінці, необхідно встановити його зв'язок із сторінкою. Можна або включити документ безпосередньо в текст сторінки, або просто зіслатися на URL XML-документа:

```
<HTML>
<HEAD>
 <TITLE>..</TITLE>
</HEAD>
<BODY>
<XML ID="dsoBook ">
<!--XML-документ -->
 </XML>
 <!-- інші елементи HTML -->
</BODY>
</HTML>
```

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

Якщо документ є коректно сформованим (при виявленні помилки відображення зупиняється без видачі повідомлення про помилку), то створюватиме об'єкт DSO. Об'єкт DSO зберігає дані XML і забезпечує доступ до них.

Для відображення зміст тегів XML-документа повинен бути зчепленим з HTML-елементом. Суть зчеплення полягає у тому, що властивості innerText тега привласнюється текст з відповідного тега XML. У табл. 7 надано елементи, які використовуються найчастіше.

Таблиця 7

HTML-елементи для зчеплення

HTML-елемент	Властивість для зчеплення елемента	Чи передає розмітку HTML?	Чи оновлює зчеплене поле XML?
A	href	Ні	Ні
BUTTON	innerHTML, innerText	Так	Ні
DIV	innerHTML, innerText	Так	Ні
IMG	src	Ні	Ні
INPUT TYPE=CHECKBOX	checked	Ні	Так
INPUT TYPE=HIDDEN	value	Ні	Так
INPUT TYPE=RADIO	checked	Ні	Так
INPUT TYPE=TEXT	value	Ні	Так
LABEL	innerText, innerHTML	Так	Ні
SPAN	innerText, innerHTML	Так	Ні
TEXTAREA	value	Ні	Так

Приклад документа:

```
<?xml version="1.0" encoding="UTF-8"?>
<DOCUMENT>Список групи
 <STUDENT>
  <FAM>Иванов</FAM>
  <NAME>Андрій</NAME>
  <photo>im1.gif</photo>
  <res> res1.doc</res>
</STUDENT>
   …
</DOCUMENT>
```

Сторінка для відображення цих даних може виглядати так:

```
<HTML>
<HEAD>
  <TITLE>Список групи</TITLE>
</HEAD>
<BODY>
<XML ID="dso2" SRC="pr6.xml"></XML>
<H2>Список групи</H2>
<TABLE>
  <THEAD>
     <TH>Прізвіще</TH>
     <TH>Ім'я</TH>
     <TH>Фото</TH>
     <TH>Резюме</TH>
 </THEAD>
  <TR ALIGN="center">
  <TD><SPAN DATASRC="#dso2" DATAFLD="FAM" ></SPAN></TD>
  <TD><SPAN DATASRC="#dso2" DATAFLD="NAME"></SPAN></TD>
  <TD><IMG DATASRC="#dso2" DATAFLD="photo"></SPAN></TD>
  <TD><A DATASRC="#dso2" DATAFLD="res">резюме</A></TD>
 </TR>
</TABLE>
<input type=BUTTON value="&lt Перша" ONCLICK="dso2.recordset.-
moveFirst()">
```

<input type=BUTTON value="&lt Попередня" ONCLICK="dso2.-recordset.movePrevious();
if (dso2.recordset.BOF) dso2.recordset.moveNext()">
<input type=BUTTON value="Наступна &gt " ONCLICK="dso2.-recordset.moveNext();
if (dso2.recordset.EOF) dso2.recordset.movePrevious()">
<input type=BUTTON value="Остання &gt" ONCLICK="dso2.record-set.moveLast()">
</BODY>
</HTML>

Пов'язання HTML-елементів дозволяє також відображати дані з XML-атрибутів. При зв'язуванні даних атрибут трактується як дочірній елемент. Наприклад, наступний запис BOOK містить атрибут з ім'ям InStock :

```
<BOOK InStock="yes ">
 <TITLE>The Adventures of Huckleberry Finn</TITLE>
 <AUTHOR>Mark Twain</AUTHOR>
 <BINDING>mass market paperback</BINDING>
 <PAGES>298</PAGES>
 <PRICE>$5.49</PRICE>
</BOOK>
```

Цей запис трактується так, як якби атрибут InStock був полем, належним BOOK, а значення InStock було б вмістом цього поля. Таким чином, елемент BOOK еквівалентний такій структурі:

```
<BOOK>
 <InStock>yes</InStock>
 <TITLE>The Adventures of Huckleberry Finn</TITLE>
 <AUTHOR>Mark Twain</AUTHOR>
 <BINDING>mass market paperback</BINDING>
 <PAGES>298</PAGES>
 <PRICE>$5.49</PRICE>
</BOOK>
```

Отже, можна використовувати звичайну техніку зв'язування даних:

```
<SPAN DATASRC="#dso1 " DATAFLD="InStock"></SPAN>
```

Змістовний модуль 2. Застосування XML у СОД

Тема 3. Застосування XML

Лабораторна робота 3. Використання мови XML у СОД

Мета: Вивчення синтаксису визначень DTD і набуття практичних навичок у їх створенні та використанні.

Лабораторне заняття забезпечує напрацювання таких ***умінь***:

розробляти DTD для XML-документів;

виконувати перевірку валідності і коректності за допомогою різних засобів.

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

обґрунтовувати вибір методів і засобів обробки XML-документів;

забезпечувати роботу з XML-документами на різних платформах.

Література: [1; 5].

Завдання лабораторної роботи

При підготовці до лабораторної роботи необхідно:

1. Опрацювати матеріал лекції, рекомендовану літературу.

2. Розробити структуру документа для використання в ході роботи, підготувати кілька варіантів документів різної структури.

При виконанні лабораторної роботи:

1. Створити DTD у тексті документа і в окремому файлі, з атрибутами і без них.

2. Виконати перевірку документів у спеціальній програмі і з використанням об'єкта DSO на HTML-сторінці двох варіантів розміщення визначень.

3. Випробувати реакцію і відображення коректних і некоректних, валідних і невалідних документів у різних браузерах, порівняти результати.

Звіт з лабораторної роботи представляється у вигляді створених документів, а також власних письмових висновків з кожного з виконаних пунктів завдання.

Контрольні запитання

1. Сформулюйте правила, що визначають коректність XML-документа.
2. Дайте визначення поняттю валідного документа.
3. Якими засобами можна перевірити валідність і коректність документа?
4. Що таке DTD, з чого вони складаються?
5. Як оголошуються типи елементів у DTD?
6. Що задають такі оголошення?

<!ELEMENT name (#PCDATA)>
<!ELEMENT title (#PCDATA | name)*>
<!ELEMENT author (#PCDATA | name)*>
<!ELEMENT article (title, author)*>
<!ELEMENT book (title, author)*>
<!ELEMENT bookstore (book | article)*>
<!ELEMENT bookshelf (book | article)*>

7. Як оголошуються припустимі атрибути елементів у DTD?

8. Що задає таке оголошення **<!ATTLIST term tC CDATA #REQUIRED>**?

Довідкові матеріали до лабораторної роботи

До пункту 1.

Під валідністю розуміється відповідність документа синтаксису, який визначає розробник. Для перевірки валідності документ повинен містити формальний опис синтаксису, визначеного розробником. Перевірка валідності можлива лише для тих документів, для яких заданий такий формальний опис. Для опису синтаксису XML-документів нині створено дві мови: DTD (Document Type Definition) і XSD (XML Schema Definition).

Оголошенням типу документа (DTD) є блок XML-розмітки. DTD може міститися в пролозі XML-документа (внутрішнє) або в зовнішньому файлі, ім'я якого вказується в документі (зовнішнє).

Внутрішнє оголошення:

...
<! DOCTYPE rootname [
<!ELEMENT rootname (contacts, issues, authors)>
...
]>
...

Зовнішнє оголошення:

<?xml version=1.0 standalone="no " ?>
<! DOCTYPE rootname SYSTEM "fileDTD.dtd">
...

Можна одночасно використовувати внутрішні і зовнішні визначення DTD:

!DOCTYPE rootname SYSTEM "URL"
[…
]>

DTD можуть містить такі типи оголошень розмітки:

оголошення типів елементів;

оголошення списків атрибутів;

оголошення компонентів.

Оголошення типу елемента має таку узагальнену форму:

<!ELEMENT Ім'я опис_вмісту>

Наприклад, оголошення типу елемента з ім'ям TITLE, для вмісту якого можуть використовуватися тільки символьні дані (дочірні елементи не допускаються).

Крім #PCDATA (parseable character data, будь-яка інформація, з якою може працювати програма-аналізатор) "опис_вмісту" може містить один з чотирьох типів вмісту: EMPTY (елемент має бути порожнім — тобто не може мати вмісту), ANY (будь-який, елемент цього типу може містити або не містити дочірні елементи, мати або не мати символьних даних), дочірній і змішаний вміст.

Якщо елемент має дочірній вміст, він може безпосередньо містити лише певні дочірні елементи, але не символьні дані. У тексті документа можна розділяти дочірні елементи символами пропуску, табуляції, повернення каретки або переведення рядка, щоб поліпшити сприйняття документа людиною, але процесор ігноруватиме ці символи і не передаватиме їх додатку.

Приклад XML-документа, що описує одну книгу:

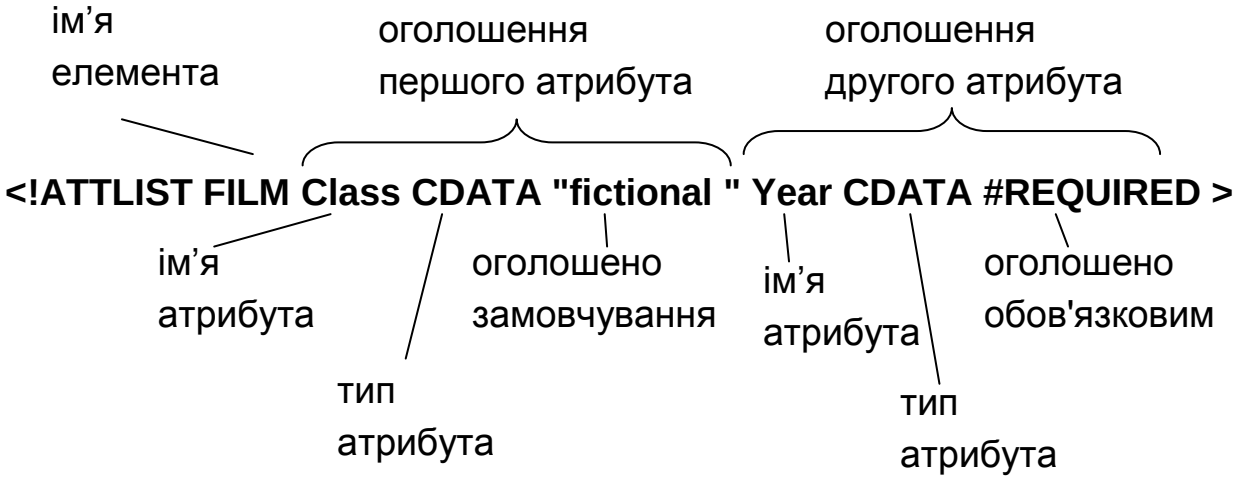
```
<?xml version="1.0" encoding="windows-1251 " ?>
<!DOCTYPE BOOK
  [
  <!ELEMENT BOOK (TITLE, AUTHOR )>
  <!ELEMENT TITLE (#PCDATA )>
  <!ELEMENT AUTHOR (#PCDATA )>
  ]
>
<BOOK >
  <TITLE>The Scarlet Letter</TITLE>
  <AUTHOR>Nathaniel Hawthorne</AUTHOR>
</BOOK >
```

Оголошення списку атрибутів має таку загальну форму:

<!ATTLIST Ім'я ВизАт>

Тут "Ім'я" є ім'я елемента, що асоціюється з атрибутом або атрибутами. "ВизАт" – це одне або декілька значень атрибутів, кожне з яких визначає один атрибут.

Приклад оголошення атрибутів:



Компонент (entity) є визначеннями, вміст яких може бути повторно використаний у документі. У мовах програмування подібні елементи називаються макровизначеннями. Створюються такі DTD-компоненти за допомогою інструкції !ENTITY:

<!ENTITY hello ' Ми раді вітати Вас!' >

Приклад документа з DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE CEL [
<!ELEMENT CEL (definition)+>
<!ELEMENT definition (#PCDATA | term)*>
<!ELEMENT term (#PCDATA)>
<!ATTLIST term   t CDATA #REQUIRED>
]>
<CEL>
<definition>Иванов
  <term t="12.00">39,6 </term>
  <term t="14.00">38,5 </term>
  <term t="16.00">36,6 </term>
</definition>
```

```
<definition>Петрова
  <term t="12.00">39,6 </term>
  <term t="14.00">38,5 </term>
  <term t="16.00">36,6 </term>
</definition>
</CEL>
```

До пунктів 2 і 3.

Документ є формально коректним, якщо він задовольняє загальні вимоги до XML-документів. При цьому перевіряється, чи немає незакритих тегів, чи всі атрибути поміщені в лапки, чи немає тегів з порушенням вкладеності, чи має документ єдиний кореневий елемент, і тому подібне. Якщо документ не є коректнім, то XML-процесор (парсер) не повинен обробляти його звичайним способом, він зобов'язаний класифікувати ситуацію як фатальна помилка. При тому подальша обробка припиняється. Так поводяться більшість браузерів. Різниця лише у повідомленнях.

За наявності DTD для перевірки валідністі можуть використовуватися програми-аналізатори, які вбудовані в більшість браузерів, модулі перевірки валідності (входить у більшість програм для роботи з XML, зокрема, в редактор XML Copy Editor та інші) та мережеві засоби.

Для отримання доступу до інформації синтаксичного розбору XML-документа в браузері IE можна використовувати такий скрипт:

```
  <HTML>
  <HEAD>
  <TITLE>Validity Tester</TITLE>
  <script LANGUAGE="JavaScript">
function pr()
  {
  xmlDoc = dsoTest.XMLDocument;
  if (xmlDoc.readyState == 4) DisplayError();
  else xmlDoc.onreadystatechange = DisplayError();
  }
  function DisplayError ()
  {
  if (xmlDoc.readyState != 4) return;
  message = "parseError.errorCode:"
```

```
+ xmlDoc.parseError.errorCode + "\n"
+ "parseError.filepos:"
+ xmlDoc.parseError.fitepos + "\n"
+ "parseError.line:"
+ xmlDoc.parseError.line + "\n"
+ "parseError.linepos:"
+ xmlDoc.parseError.linepos + "\n"
+ "parseError.reason:"
+ xmlDoc.parseError.reason + "\n"
+ "parseError.srcText:"
+ xmlDoc.parseError.srcText + "\n"
+ "parseError.url:"
+ xmlDoc.parseError.url;
alert (message);
}
</script>
</HEAD>
<BODY onLoad="pr();">
<!--Встановити значення SRC, рівне URL XML-документа, який треба перевірити -->
<XML ID="dsoTest" SRC="pr3.xml"></XML>
<h1>для перевірки валідності записати ім'я файла і натиснути "Обновить" у браузері</h1>
</BODY>
</HTML>
```

Мережеві засоби:

http:// validator.w3.org/ – офіційний модуль перевірки HTML-коду консорціуму W3C;

http://www.xml.eom/pub/a/tools/ruwf/check.html – модуль перевірки дійсності XML-коду групи XML.COM, заснований на процесорі Lark;

www.ltg.ed.ac.uk/~richard/xml-check.html – модуль перевірки XML-коду групи Language Technology Group Единбурзького університету, заснований на синтаксичному аналізаторі RXP;

http://www.stg.brown.edu/service/xmlvalid/ – модуль перевірки дійсності XML-коду групи Scholarly Technology Group при університеті Брауна.

Тема 4. Обробка XML-документів

Лабораторна робота 4. Застосування XSLT для обробки XML-документів

Мета: дослідження можливостей розширюваної мови таблиць стилів (XSL) для обробки XML-документів.

Лабораторне заняття забезпечує напрацювання таких ***умінь***:

створювати XSLT для перетворень і обробки XML-документів;

створювати найпростіші XSL-FO документі;

перетворювати XSL-FO документи в інші формати.

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

організація обробки даних з XML-документів шляхом їх трансформації;

автоматизація процесу публікації даних з XML-документів.

Література: [1 – 3].

Завдання лабораторної роботи

При підготовці до лабораторної роботи необхідно:

1. Опрацювати матеріал лекції, рекомендовану літературу і матеріали, видані викладачем.

2. Підготувати документи для роботи (самостійно розроблені XML-документи та кілька варіантів таблиць XSLT для перетворень).

3. Продумати і підготувати оригінальне оформлення для результуючих документів.

При виконанні лабораторної роботи:

1. Використання XSLT для перетворення XML-документів.

1.1. Створити і випробувати перетворення XML-документа в HTML у браузері і редакторі.

1.2. Створити і випробувати перетворення XML-документа в інший XML-документ у браузері і редакторі.

2. Використання XSLT для обробки XML-документів.

2.1. Створити і випробувати перетворення XML-документа з фільтрацією даних у браузері і редакторі.

2.2. Створити і випробувати перетворення XML-документа з сортуванням даних у браузері і редакторі.

3. Використання документів XSL-FO.

3.1. Створити документ XSL-FO, який описує макет одної зі сторінок (погодити з викладачем).

3.2. Створити XSLT для перетворення одного з раніше створених документів в XSL-FO.

3.3. Перетворити створені документі XSL-FO в формат pdf.

Звіт з лабораторної роботи представляється у вигляді створених документів.

Контрольні запитання

1. Для чого служить мова XSL?
2. Сформулюйте загальні правила завдання XSL-перетворень.
3. Що таке шаблон в XSLT?
4. Назвіть відомі вам XSL-елементи.
5. Які перетворення задає наступна група елементів?

```
<xsl:for-each select="список/источник">
 <LI><SPAN STYLE="font-style:italic">
  <xsl:value-of select="автор"/>
  <xsl:value-of select="название"/>
  <xsl:value-of select="издательство"/>
  <xsl:value-of select="год"/>
  </SPAN>
  <xsl:value-of select="кол_страниц"/></LI>
</xsl:for-each>
```

6. Дайте визначення форматуючого об'єкта.
7. Опишіть структуру XSL-FO-документа.
8. Дайте опис майстер-сторінки в XSL-FO.
9. Як описуються дані, що розміщуються на сторінках?
10. Що таке властивості форматуючих об'єктів? Для чого вони використовуються?
11. Як створюються XSL-FO-документи і які можливості вони надають?

Довідкові матеріали до лабораторної роботи

До пункту 1.

XSLT – це XML-додаток, що визначає правила, відповідно до яких один XML-документ перетвориться в іншій. XSLT-документ, тобто трансформуюча таблиця стилів XSLT, містить шаблони (опис форматування). XSLT-процесор порівнює елементи вхідного XML-документа із шаблонами в таблиці стилів. Коли відповідний шаблон знайдений, процесор записує вміст шаблону у вихідне дерево. Після закінчення цього етапу вихідне дерево записується або в XML-документ, або в інший формат, такий, як звичайний текст або HTML.

XSLT-документ може виглядати так:

<?xml version="1.0"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<!-- один або декілька елементів шаблону...-->

</xsl:stylesheet >

Файли, що містять XSLT, мають розширення xsl. XML-документи, що підлягають обробці з використанням таблиць стилів, повинні мати в пролозі інструкцію обробки xml-stylesheet, яка указує, де знаходиться відповідна документу таблиця стилів наприклад:

<?xml-stylesheet type="text/xml" href="http://www.ore.com/styles/people.xsl"?>

Кожен шаблон представлений елементом xsl:template. Цей елемент має атрибут match, який вказує на певну гілку дерева структури документа (атрибут match аналогічний селектору в правилі CSS). Наприклад, шаблон, який містить інструкції для відображення всього XML-документа:

<xsl:template match="/">

<!-- дочірні елементи... -->

</xsl:template >

Інший простий зразок – це ім'я елемента, наприклад, <xsl:template match="person">Людина</xsl:template>. Цей шаблон говорить про те, що кожного разу, коли зустрічається елемент person, процесор таблиць стилів повинен генерувати текст "Людина".

Кожна XSL-таблиця стилів повинна містити тільки один шаблон із атрибутом match, який має значення "/". Крім того, можна також включити один або декілька додаткових шаблонів з інструкціями для відображення певних підлеглих гілок у структурі XML-документа. Для активізації останніх слід використовувати спеціальні елементи.

Шаблон може містити послідовність елементів двох видів:

XML-елементи, що представляють незмінну коректну розмітку (наприклад, HTML-розмітку, а може бути яку-небудь іншу, в яку виконується перетворення);

XSL-елементи, які задають порядок використання даних із початкового документа (інструкції по вибору і модифікації даних XML).

Приклад вставки незмінної розмітки:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template match="/">
        <HTML >
        <HEAD >
        <TITLE>pr10</TITLE>
        </HEAD >
        <BODY >
        <H2>Результат перетворення</H2>
        </BODY >
        </HTML >
 </xsl:template >
</xsl:stylesheet >
```

Для зміни порядку використання даних і обробки існують спеціальні елементи (елементи трансформації). Деякі елементи для перетворення наведені в табл. 8.

Таблиця 8

Елементи трансформації

Елемент	Призначення	Атрибути
<xsl:template>	опис шаблону	match
<xsl:apply-templates>	вставка даних за шаблоном	select
<xsl:value-of>	вставка значень з елемента	select
<xsl:for-each>	повторення обробки	select, order-by
<xsl:sort>	сортування	select, order
<xsl:attribute>	установка значень атрибутів	name

Далі наведено приклад XML-документа і XSLT для його перетворення у HTML.

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="pr.xsl" ?>
<PRODUCTS> Товары
   <PRODUCT import="yes">
      <NAME> Product #1 </NAME>
      <SORT> Первый
         <COLOR> red </COLOR>
         <PRICE> $10.00 </PRICE>
      </SORT>
      <SORT>Второй
```

```
            <COLOR> blue </COLOR>
            <PRICE> $11.00 </PRICE>
          </SORT>
          <SORT>Третий
            <COLOR> gray </COLOR>
            <PRICE> $16.00 </PRICE>
          </SORT>
        </PRODUCT>
        <PRODUCT ID="i1">
          <NAME> Product #2 </NAME>
          <SORT>4
            <COLOR> red </COLOR>
            <PRICE> $20.00 </PRICE>
          </SORT>
          <SORT> 5
            <COLOR> green </COLOR>
            <PRICE id="i2"> $22.00 </PRICE>
          </SORT>
        </PRODUCT>
      </PRODUCTS>
```

```
      <?xml version="1.0"?>
      <xsl:stylesheet                                          version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:template match="/">
            <HTML>
            <HEAD>
            <TITLE>ПРОДУКТЫ</TITLE>
            </HEAD>
            <BODY>
            <H2>Список продуктов на складе №1</H2>
            <xsl:apply-templates select="PRODUCTS/PRODUCT"/>
            </BODY>
            </HTML>
        </xsl:template>
        <xsl:template match="PRODUCT">
            <SPAN STYLE="font-style:italic">Наименование: </SPAN>
            <SPAN><xsl:value-of select="NAME"/></SPAN>
            <SPAN STYLE="font-style:italic">Импортный: </SPAN>
            <SPAN><xsl:value-of select="./@import"/></SPAN>
            <BR/>
```

```
        <xsl:apply-templates select="SORT"/>
        <BR/>
    </xsl:template>
    <xsl:template match="SORT">
    <SPAN><xsl:value-of select="COLOR"/></SPAN>
    <SPAN><xsl:value-of select="PRICE"/></SPAN>
    <BR/>
    </xsl:template>
</xsl:stylesheet>
```

До пункту 2.

Для сортування у просторі імен Transform використовується спеціальний елемент (інструкція) sort, що розміщується безпосередньо після for-each і apply-templates:

```
<xsl:sort select = "строкове_вираження "
 data-type = "text "або "number "
 lang = "код_мови "
 order = "ascending " або "descending "
 case-order = "upper-first " або "lower-first " />
```

Цей елемент змінює порядок проходження вузлів з порядку, прийнятого в документі, на інший, наприклад, за абеткою.

Фільтр – це вираз, поміщений в квадратні дужки ([]), який слідує безпосередньо за послідовністю вузлів шляху. Наприклад, зразок, привласнений наступному атрибуту match, указує, що відповідний елемент повинен носити ім'я BOOK, і крім того, повинен мати дочірній елемент BINDING, який містить текст "trade paperback":

```
<xsl:template match="BOOK [BINDING='trade paperback ']">
```

Для виконання фільтрації можна також використовувати елемент <xsl:if test="number(nam)>10">…</xsl:if> Вкладені в цей тег елементи відображаються тільки тоді, якщо обчислена умова істинна.

До пункту 3.

Мова XSL-FO призначена для опису зовнішнього вигляду документа. На противагу CSS, XSL-FO — це XML-додаток, призначений для опису точного розміщення тексту на сторінці. Це інструмент верстки.

Файли зазвичай мають розширення .fo. Ці документи призначені для програм, які перетворять (конвертують) їх в документи кінцевого формату (для конкретних пристроїв, наприклад, PDF для принтера або HTML для браузера).

Імена всіх об'єктів документа мають префікс fo. Кореневий елементі fo:root. Дочірніми елементами fo:root є один елемент fo:layout-master-set і один або декілька елементів fo:page-sequence.

Елемент fo:layout-master-set містить шаблони створюваних сторінок, а в елементах fo:page-sequence розміщується основний вміст документа (тексти і зображення). У загальному випадку структура документа виглядає так:

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
<fo:root >
 <fo:layout-master-set >
 <fo:simple-page-master master-name="first ">
 <fo:region-body/>
 </fo:simple-page-master >
 </fo:layout-master-set >
 <fo:page-sequence master-name="first ">
 <!-- дані, що поміщаються на сторінці, ->
 </fo:page-sequence >
</fo:root >
```

Кожен елемент fo:simple-page-master задає загальний макет сторінки, включаючи розмір областей: before region, body region, after region, end region і start region. На рис. 9 наведено розташування цих областей.

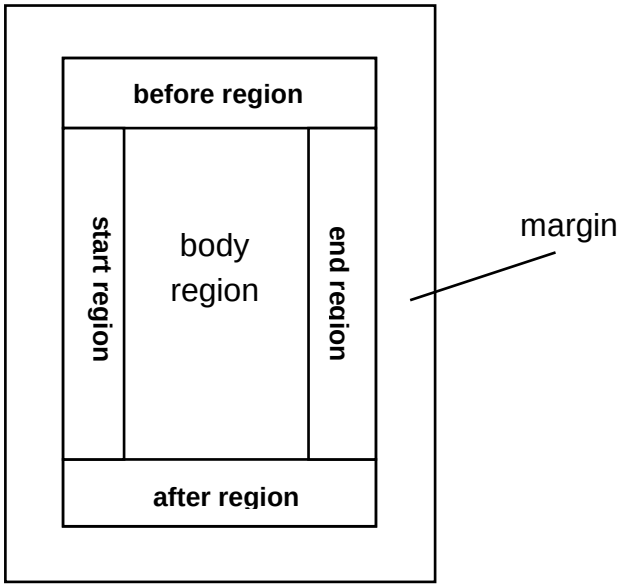


Рис. 9. **Типовий макет сторінки**

У кожній області може розміщуватися свій вміст. За допомогою атрибутів елемента fo:simple-page-master задаються:

master-name – ім'я, яке використовується для зв'язку з вмістом;

page-height – висота сторінки;

page-width – ширина сторінки;

margin-bottom, margin-left, margin-right і margin-top, або узагальнений атрибут margin — відступи;

writing-mode – напрям, у якому йде текст на сторінці, наприклад, left-to-right (зліва направо), right-to-left (справа наліво) або top-to-bottom (зверху вниз);

reference-orientation – поворот вмісту сторінки з кроком у 90 градусів.

Розміри областей, а також відстані між ними задаються в дочірніх елементах fo:simple-page-master (кожен описує свою область): fo:region-before, fo:region-after, fo:region-body, fo:region-start, fo:region-end.

Приклад завдання елемента fo:layout-master-set з однією майстер-сторінкою:

```
<fo:layout-master-set >
 <fo:simple-page-master master-name="only "
 page-width="8.5in" page-height="11in "
 margin-top="0.5in" margin-bottom="0.5in"
 margin-left="0.5in" margin-right="0.5in">
 <fo:region-start extent="1.0in"/>
 <fo:region-before extent="1.0in"/>
 <fo:region-body margin="1.0in"/>
 <fo:region-end extent="1.0in"/>
 <fo:region-after extent="1.0in"/>
 </fo:simple-page-master >
</fo:layout-master-set >
```

Крім елемента fo:layout-master-set, кожен документ форматуючих об'єктів включає один або декілька елементів fo:page-sequence, що описують розміщуваний вміст. Кожен елемент page-sequence пов'язаний з майстер-сторінкою, яка визначає макет розміщення даних. Для зв'язку використовується атрибут master-reference, значенням якого є ім'я однієї з майстер-сторінок в елементі fo:layout-master-set.

Елемент з fo:page-sequence може містити декілька дочірніх елементів:

необов'язковий елемент fo:title (його можна використовувати як титул документа, наприклад, так, як це робить елемент TITLE в HTML);

декілька необов'язкових елементів fo:static-content (вони містять текст, який повторюватиметься на кожній сторінці);

декілька елементів fo:flow (для різних областей), які містять основні дані, що займають послідовність з декількох сторінок.

Елемент fo:static-content містить інформацію, яка повинна розташовуватися на кожній сторінці. Наприклад, з його допомогою у верхню частину кожної сторінки можна поміщати назву книги або розділу. Статичний вміст може розміщуватися в різних областях майстер-сторінки. Наприклад, назва розділу може розміщуватися на лівих сторінках розворотів, а назва книги — на правих.

Елемент fo:flow іноді позначають терміном потік, він містить елементи, що послідовно розміщуються на сторінці (блоки тексту, списки, рисунки і так далі).

Вміст потоку (елемента fo:flow) складається з послідовності елементів fo:block (блок тексту), fo:table (таблиця) і fo:list-block (список) і так далі.

Для вказівки області, в якій розміщуватимуться елементи потоку, використовується атрибут flow-name елемента fo:flow. Допустимі значення: xsl-region-body, xsl-region-before, xsl-region-after, xsl-region-start, xsl-region-end. Сенс цих значень очевидний.

У одній і тій же послідовності сторінок не може бути двох потоків з одним і тим же ім'ям. Таким чином, кожен елемент fo:page-sequence може містити до п'яти дочірніх елементів fo:flow (поодинці для кожної області сторінки).

Елементи fo:static-content, якщо вони є, повинні розташовуватися перед елементами fo:flow.

Приклад документа з форматуючими об'єктами:

```
<?xml version="1.0"  encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
 <fo:layout-master-set >
    <fo:simple-page-master master-name="A4"
                                  page-width="210mm " page-height="297mm "
                                  margin-top="0.5in" margin-bottom="0.5in"
                                  margin-left="0.5in" margin-right="0.5in">
      <fo:region-body margin-top="1.5in"  margin-bottom="1.5in"/>
       <fo:region-before extent="1.0in"/>
       <fo:region-after extent="1.0in"/>
 </fo:simple-page-master >
 </fo:layout-master-set >
 <fo:page-sequence font-family="Times New Roman" master-
reference="A4"  initial-page-number="1">
 <fo:static-content flow-name="xsl-region-before">
 <fo:block  font-size="12pt" font-weight="bolder">  Засоби систем обробки
даних </fo:block>
 </fo:static-content >
 <fo:static-content flow-name="xsl-region-after">
 <fo:block font-size="16pt" font-weight="bolder">стр. <fo:page-
number/></fo:block>
 </fo:static-content >
 <fo:flow  font-size="16pt" flow-name="xsl-region-body">
```

```
<fo:block> Заголовок</fo:block>
<fo:block  font-size="20pt"> Засоби систем обробки даніх</fo:block>
<fo:block break-before="page"/>
<fo:block> … текст ...</fo:block>
</fo:flow >
</fo:page-sequence >
</fo:root >
```

Перетворення у кінцевий формат може здійснюватися за допомогою програми RenderX.

Робота з програмою RenderX.

Після інсталяції може знадобитися деяке налаштування, яке полягає у вказівці шляхів до різних компонентів, використовуваних при перетворенні (бібліотеки, тимчасові файли, шрифти і т. п.). Велика частина налаштувань може бути залишена за замовчуванням. Потрібно тільки задати доступ до шрифтів, що містить кирилицю. Для цього необхідно відкрити файл конфігурації xep.xml, знайти блок, який починається з коментаря

<!-- Sample configuration for Windows TrueType fonts. -->.

За цим блоком слідують теги, що містять інформацію про кириличні шрифти ОС Windows (Arial, Times New Roman, Courier New, Tahoma, Verdana, Palatino), Вони поміщені в коментар. Символи коментарів необхідно прибрати і шрифти стануть доступні для налаштування в закладці Fonts команди Edit із меню Options. Для всіх шрифтів слід включити режим true перемикачем Embedded. Якщо потрібно зробити доступним шрифт, якого немає в цьому списку, то інформацію про нього слід помістити в конфігураційний файл, використовуючи наявні теги в якості зразка. При необхідності використовувати родове ім'я (наприклад, serif), слід вибрати пункт Windows True Type, і у вікні, додати в таблицю замін (кнопка Add Alias) список шрифтів.

Перетворення FO-документів можна виконувати як з командного рядка (xep.bat), так і за допомогою графічної оболонки Assistent (x4u.bat).

Вікно графічної оболонки забезпечує дії з відкриття файла .fo (меню File), перетворення (меню Formatting) і налаштування (меню Options). Саме перетворення виконується командою Start з меню Formatting, у вікні, що відкривається, вибирається формат вихідного документа (некомерційна версія підтримує тільки PS і PDF, за замовчуванням PDF). Після успішного перетворення створюється файл з результатом і у вікні Event Log виводиться інформація про виповнені дії. При неможливості виконання перетворень в цьому вікні знаходиться інформація про помилки.

Рекомендована література

Основна

1. Молчанов В. П. Засоби систем обробки даних : навч. посібн. / В. П. Молчанов. – Х. : Вид. ХНЕУ, 2012. – 103 с.

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

Додаткова

3. Гарольд Э. XML : справочник / Э. Гарольд, С. Минс. – СПб. : Символ-Плюс, 2002. – 576 с.

4. Навврро Э. XHTML : учебный курс / Э. Навврро. – СПб. : Питер, 2001. – 336 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.

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Методичні рекомендації
до виконання лабораторних робіт
з навчальної дисципліни
"ЗАСОБИ СИСТЕМ ОБРОБКИ ДАНИХ"
для студентів галузі знань 0515
"Видавничо-поліграфічна справа"
всіх форм навчання

Укладач **Молчанов** Віктор Петрович

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

Редактор **Бутенко В. О.**

Коректор **Мартовицька-Максимова В. А.**

План 2013 р. Поз. № 192.
Підп. до друку Формат 60×90 1/16. Папір MultiCopy. Друк Riso.
Ум.-друк. арк. 2,5. Обл.-вид. арк. 3,13. 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.

Методичні рекомендації
до виконання лабораторних робіт
з навчальної дисципліни
"ЗАСОБИ СИСТЕМ ОБРОБКИ ДАНИХ"
для студентів галузі знань 0515
"Видавничо-поліграфічна справа"
всіх форм навчання