# ЛАБОРАТОРНА РОБОТА №1

**Тема:** Створення XML-документу і його відображення з допомогою каскадних таблиць стилів CSS.

**Мета роботи:** розглянути структуру XML-документу, та виконати створення власного XML-документ згідно з правилами створення well-formed XML- документу. Вивчити основи застосування CSS при формуванні XML- документу. Навчитися відображати XML-документи з використанням каскадної таблиці стилів

# ТЕОРЕТИЧНІ ВІДОМОСТІ

XML - текстовий формат, призначений для зберігання структурованих даних (замість існуючих файлів баз даних), для обміну інформацією між програмами, а також для створення на його основі більш спеціалізованих мов, іноді званих словниками. XML, які є спрощеною підмножиною мови SGML. Стандартом визначено два рівні правильності документа XML:

Опис XML- документа являє собою простий текст, який можна набрати в будь-якому текстовому редакторі. Наприклад, створіть у текстовому редакторі Notepad новий файл і введіть текст XML- документа та збережіть його з розширенням .xml

Кожен XML- документ розмічається тегами. Тег – це текст, укладений в кутові дужки, який не відноситься до змісту документа, а вказує на початок або кінець будь-якого документа.

***Елемент документа XML*** має початковий тег, що задає ім'я елементу і додаткову інформацію (атрибути), і кінцевий тег, що містить таке ж саме ім'я елемента зі знаком "/" попереду. Елементи визначають логічну структуру документа і несуть в собі інформацію, що міститься в документі. Ім'я елемента вважається так само його типом. Елементи XML-документа можуть бути вкладеними.

Поки не заданий формат відображення XML- документа на екрані, браузер застосовує спосіб, прийнятий за замовчуванням (IE5 буде використовувати вбудовану таблицю стилів для відображення документа). Одним з варіантів вказівки способу відображення документа є створення для нього таблиці каскадних стилів (CSS).

***Правильно оформленими (well-formed) XML-документом*** називається документ, що задовольняє мінімальному набору правил відповідності для XML- документа. *Правильно оформлений XML-документ складається з двох основних частин:* ***прологу і кореневого елемента.*** Крім цього він може містити ***коментарі, інструкції і пропуск***. Кореневий елемент може містити вкладені елементи. Елементи повинні бути правильно вкладені. Якщо елемент розпочинається всередині деякого іншого елемента, то й закінчуватися він повинен всередині того ж елемента.

Кожен елемент складається з початкового тега, вмісту і кінцевого тега. Винятком є порожній елемент, який може використовувати єдиний тег порожнього елемента <Emptylement />.

## Правила використання імен елементів (типів елемента):

* ім'я повинне починатися з букви або символу підкреслення (\_),
* наступні після першого символи можуть бути буквами, цифри, точкою, тире або підкресленням.
* Не слід використовувати імена, що починаються із префікса "xml" (в будь-якому поєднанні малих або великих літер).
* ім'я, що записане в початковому тегу, має в точності відповідати імені в кінцевому тезі;
* Дотримання регістра істотно для імен елементів, як і для всього тексту в описі розмітки.

***Вмістом елемента*** вважається текст, розташований між початковим і кінцевим тегами. У початковий тег елемента або в тег порожнього елемента можна включити один або кілька описів атрибутів. Опис атрибута являє собою пару ***ім'я***

***= значення***, пов'язану з даним елементом. Кожне ім'я атрибута може тільки один раз бути присутнім в початковому тегу елемента. Правила іменування атрибутів аналогічні правилам іменування елементів. Завдання атрибутів забезпечує альтернативний спосіб включення інформації в елемент. Значення, яке можна привласнювати атрибуту, є рядком символів, обмежених одинарними або подвійними лапками.

***Коментар*** починається з символів ***<! –*** і закінчується символами ***–>***. Усередині коментарів не може міститися подвійне тире (--), символ лівої кутової дужки (<) і знак амперсанда (&). Коментарі можна вставляти в будь-яке місце XML-документа, крім опису тега.

***Інструкції*** (інструкції по обробці) призначені для XML-процесора, який буде обробляти документ. Загальний вигляд інструкції:

## <? найменування дані?>

***Найменування*** вказує одержувача інструкції. Параметр ***дані*** задає зміст інструкції. За назвою конкретний XML-процесор визначає, призначені дані йому або іншому процесору. Інструкції можуть поміщатися в будь-яке місце XML- документа поза описом тегів.

*Наприклад, інструкція*

<?Xml-stylesheet type="text/css"href="file\_2.css"?>

наказує Internet Explorer використовувати CSS-таблицю з файлу file\_2.css.

Усередині символьних даних у вмісті елементу не можна поміщати деякі спеціальні символи (<, &, ...), так як це може призвести до плутанини при обробці документа. Одним з можливих шляхів подолання цих обмежень є використання розділів символьних даних. Такий розділ починається з символів ***<! [CDATA [і закінчується]]>.***

Всі символи всередині розділу CDATA розглядаються як літеральна частина символьних даних елемента, а не як XML-розмітка. Розділ CDATA може розташовуватися в будь-якому місці документа, що займається символьними даними. Розділи CDATA не можуть бути вкладеними.

Документ XML може містити ***порожні рядки***, що складаються з одного або декількох пропусків, символів табуляції, символу Enter. Можна вільно додавати прогалини і переклади рядків між: початковими і кінцевими тегами; коментарями; інструкціями по обробці.

Неповинно бути пропусків між кутовою дужкою що відкривається і іменем о елемента.

# ПРИКЛАД ВИКОНАННЯ РОБОТИ

*Розглянемо структуру XML*

Перший рядок XML-документа називається ***оголошення XML –*** це необов'язковий рядок, який вказує версію стандарту XML (зазвичай це 1.0), також тут може бути вказано кодування символів і зовнішні залежності. Наприклад:

<? Xml version = "1.0" encoding = "utf-8"?>

Найважливіша обов'язкова синтаксична вимога полягає в тому, що документ має ***тільки один кореневий елемент*** (так само іноді званий *елемент документа*). Це означає, що текст або інші дані всього документа повинні бути розташовані між єдиним початковим кореневих тегом і відповідним йому кінцевим тегом. Наприклад:

<Comp> комп'ютер </comp>

Інша частина цього XML-документа складається з вкладених елементів, деякі з яких мають атрибути і вміст. ***Елемент*** зазвичай складається з тегів, що відкриваються і закриваються, обрамляють текст і інші елементи.

<Video> відеокарта </video>

Крім змісту у елемента можуть бути ***атрибути*** – пари *ім'я–значення*, що додаються в тег, що відкривається після назви елемента. Значення атрибутів

завжди розташовані в лапки (одинарні або подвійні), одне і те ж ім'я атрибута не може зустрічатися двічі на одному елементі. Не рекомендується використовувати різні типи лапок для значень атрибутів одного тега. Наприклад:

<Video model = "6600"> відеокарта </video>

У наведеному прикладі у елемента *«video»* є атрибут: *«model»*, що має значення *«6600»*

У будь-якому місці дерева може бути розміщений елемент-коментар. XML- коментарі розміщуються всередині спеціального тега, який починається з символів

*<! –* і закінчується символами *–>*. Два знака дефіс (--) у коментарі присутнім не можуть бути.

<!-- Коментар-->

*Наприклад:*

<log>

<event date=" 27/May/2016:04:22:41 " result="success">

<ip-from> 192.11.112.8 </ip-from>

<method>GET</method>

<url-to> /misc/</url-to>

<response>200</response>

</event>

</log>

# ЗАВДАННЯ

Розглянемо приклад виконання завдання. Необхідно створити Xml-документ на основі даної таблиці.

Використовувати виділені слова лівого стовпчика таблиці, як теги, а звичайні

– як атрібути. В якості параметрів використовуйте дані правого стовпчика. Збережіть документ з назвою Завдання1.xml

|  |  |
| --- | --- |
| **Основні** | |
| Виробник | ASUS |
| Модель | EAH5670/DI/512MD5 |
| Длина | 170 mm |
| **Відео** | |
| Відеопам’ять | 512 Мб |
| RAMDAC | 400МГц, 10 бит на канал |
| Максимальна роздільна здатність 2D/3D | 2560x1600 або 1920x1200 для HDMI |
| Підтримка HDCP | 1080p |
| Підтримка API | DirectX11, OpenGL 3.2 |
| **Підтримка пам’яті** | |
| Тип пам’яті | GDDR5 |
| Розрядність шини пам'яті | 128бит |
| Частота пам'яті | 1000Гц |
| **Конфігурація** | |
| Чіп | Radeon HD 5670 |
| Техпроцес | 40HM |

|  |  |
| --- | --- |
| Частота чипу | 775 МГц |
| Кількість піксельних конвейерів | 20,8 блоков выборки текстур |
| Кількість процесорів | 4 |
| **Роз'єми і виходи** | |
| Порти | DVI-I, HDMI,15-піновий конвектор D-Sub |
| **Інтерфейс** | |
| Інтерфейс | PCI Express 2.116x |
| **Охолодження** | |
| Управління швидкістю обертання | Немає |
| Охолодження | Активне |
| Конструкція системи охолодження | Система охолодження сусідній з відеокартою слот |
| **Живлення** | |
| Роз'єми живлення | Немає |
| **Сумісність** | |
| Вимоги до системи | Блок живлення 400 Вт або 500Вт |
| Підтримка обчислень загального призначення | ATI Stream, OpenCL 1.0, DirectCompute 11 |
| Підтримка ОС | Windows XP, Windows 7 |
| **Логістика** | |
| Розміри упаковки | 30.5x22.5x5.7 |
| Вага бруто | 0.755 кг |

В результаті Ваш створених XML-файл може виглядати наступним чином:

<?xml version="1.0" encoding="utf-8"?>

<Harakter>

<Osnovnie virobniki="ASUS " model="EAH6670/DI/512MD5 " dlina=" 170 мм ">

</Osnovnie>

<Video videopamyat="512" RAMDAC="400 МГц, 10 бит на канал." HDCP="Есть (1080p)" API="DirectX 11 и OpenGL 3.2 ">

</Video>

<Pamyat tip="ddr5" shina="128 біт" chastota="1000 МГц">

</Pamyat>

<Config chip="Radeon HD 5670" chastota\_chipa="775 МГц" kol- vo\_processorov="4 ">

</Config>

<Razemi porti="D-sub,dvi">

</Razemi>

<Interface>PCI Express 2.1</Interface>

<Ohlazhdenie upravlenie="net" ohlazhdenie="activ">

</Ohlazhdenie>

<Sovmestimost pitanie="400 Вт" os="Windows XP x64, Windows7 >

</Sovmestimost>

</Harakter>

**ВАРІАНТИ ЗАВДАНЬ:**

За наданою схемою на рисунках і обраного варіанту необхідно створити XML-документ. Для кожного елемента першого рівня передбачити не менше 3 примірників, для кожного елемента наступних рівнів – не менше двох примірників.

**Варіант №1** Результати сесії студентів

Успішність

Дисципліна

Спеціальність

Група

Студент

Код Назва

Цикл Звітність

Код

Назва

Номер Курс

Код

Номер залік. книжки

ПІБ Дата

Складання

Оцінка

**Варіант №2** Книжковий каталог

Книга

Жанр

Автор

Код Назва

Рік видання Кількість сторінок

Код

Назва

Код ПІБ

Розташування Видавництво

Код Назва

Код Назва

**Варіант №3** Анонс репертуару кінотеатру

Кінотеатр

Фільм

Жанр Режисер

Сценарист

Актор

Код Назва

Рік видання

Код

Назва Код

ПІБ

Код

ПІБ

Код

ПІБ

**Варіант №4.** Інформаційне табло аеропорту

Аеропорт

Рейс Номер

Літак Серія

Номер

Авіакомпанія

Пасажир

Код Назва

Код

ПІБ

Паспорт

Маршрут

Місце

Код

Код

Номер Клас

Відправлення Прибуття

Назва Код Назва

**Варіант № 5.** Облік співробітників фірми в відділі кадрів

Відділ кадрів

Підрозділ

Код

Назва

Співробітник

Код ПІБ

Стать

Дата народження

Дата зарахування у фірму

Код

Посада

Категорія

Кваліфікація

Назва

Код

Назва Код

Назва

**Варіант № 6.** Записна книжка адресів **Варіант № 7.** Розклад занять в університеті **Варіант № 8.** Облік товарів на складі **Варіант № 9.** Відділ кадрів студентів

**Варіант № 10.** Каталог загальноосвітніх Інтернет-ресурсів

Відображення XML-документів з використанням каскадної таблиці стилів здійснюється в два етапи: створення файлу таблиці стилів і зв'язування таблиці стилів з XML-документом.

Таблиця стилів складається з одного або декількох правил, іноді їх називають набором правил. Правило містить інформацію по відображенню певного типу елемента в XML-документі. Селектор являє собою ім'я типу елемента, до якого відноситься інформація для відображенню.

Каскадні таблиці стилів CSS не чутливі до регістру. Набір успадкованих властивостей, які присвоєні певному елементу, діє на всі дочірні елементи, прямо або побічно вкладені в нього, якщо тільки вони не переустановлюються згодом для певного дочірнього елемента.

Для неуспадкованих властивостей, якщо не задано значення властивості для конкретного елемента, браузер використовує значення властивості за замовчуванням.

## Використання контекстуальних селекторів

***Контекстуальним селектором (contextual)*** називається селектор, в якому ім'я елемента може передувати іменам одного або декількох елементів-предків (батьківський, батьківський плюс батьківський батька і т.п.). У цьому випадку правило буде застосовано тільки до елементів з цим ім'ям, які є вкладеними подібним чином. Між іменами елементів в контекстному селекторі ставлять тільки пробіли.

***Родовим селектором (generic)*** називається селектор, який не включає імен елементів-предків.

Якщо певна властивість для одного і того ж елемента має одну установку в правилі з контекстуальних селектором, і іншу установку в правилі з родовим селектором, установка в правилі з контекстуальних селектором домінує, оскільки є більш конкретизована.

Наприклад, фрагмент програмного коду XML-документа, де <MAPS> – кореневий елемент:

<MAPS>

<CITY>

<STATE>California</STATE>

</MAPS>

Правила приєднаної таблиці стилів: CITY STATE

{font-style:normal} STATE

font-style:italic}

<NAME>Santa Fe</NAME>

<STATE>New Mexico</STATE>

</CITY>

В результаті браузер відобразить «New Mexico» звичайним шрифтом, а

«California» – курсивом.

## Використання атрибуту STYLE

Можна в XML-документі використовувати спеціальний атрибут STYLE, замість того, щоб встановлювати одне або кілька певних властивостей окремого елемента в таблиці стилів.

Якщо значення властивості, встановленої за допомогою атрибута STYLE, конфліктує із значенням властивості, встановленого в таблиці стилів, установка за допомогою атрибута STYLE має пріоритет. Таким чином, атрибут STYLE є зручним засобом, щоб перевстановити для певного елемента значення властивості, присвоєне для типу елемента в приєднаної таблиці стилів.

Використання атрибуту STYLE порушує принцип CSS щодо зберігання інформації про форматування окремо від визначення вмісту документа і структури XML-файла. Щоб встановити одне або кілька значень властивостей, оголошення в значення атрибуту STYLE включається у вигляді рядка, укладеного в лапки, відокремлюючи індивідуальні оголошення крапкою з комою. Наприклад,

<TITLE STYLE='font-style:normal;

font-size:14pt'> The Adventures of Huckleberry Finn TITLE/>

## Імпорт інших таблиць стилів

Можна за допомогою директиви @import в одну каскадну таблицю стилів вбудувати одну або кілька інших таблиць стилів. Можливість імпорту окремих таблиць стилів дозволяє зберігати правила для пов'язаних стилів в окремих файлах, а потім об'єднувати їх при створенні документів певного типу. Директива @import

повинна розташовуватися на початку таблиці стилів перед правилами. На початку таблиці стилів можна розмістити кілька директив @import.

@import url (URL ІмпортТаблСтіл);

Наприклад

/\* Файл: Inventory01.css \*/ @import url (Book.css); BOOK

{Display: block; margin-top: 12pt; font-size: 10pt}

/\* Продовження таблиці стилів ... \*/

У разі виникнення конфлікту правил основна таблиця стилів (з файлу, в який здійснюється імпорт) має пріоритет над імпортованими таблицями стилів. Якщо імпортуємо кілька таблиць стилів, правила з таблиці стилів, імпортованої останньою, мають пріоритет над правилами з раніше імпортованих таблиць стилів.

## Зв'язування таблиці стилів з XML-документом

Щоб зв'язати таблицю каскадних стилів з XML-документом, необхідно вставити в документ зарезервовану інструкцію по обробці xml-stylesheet. Ця інструкція з обробки має наступну узагальнену форму записи, де URLТаблСтіл є URL, що задає місцезнаходження файлу таблиці стилів:

<? Xml-stylesheet type = "text / css" href = "URLТаблСтіл"?>

Можна використовувати повний URL, наприклад:

<? Xml-stylesheet type = "text / css" href = "http://www.my\_domain.com/Inventory01.css"?>

Найчастіше використовується частковий URL, який задає місцезнаходження

щодо місцезнаходження XML-документа, що містить інструкцію з обробки xml- stylesheet, наприклад:

<? Xml-stylesheet type = "text / css" href = "file\_2.css"?>

Зазвичай інструкція по обробці xml-stylesheet додається в пролог XML- документа, слідом за оголошенням XML. Якщо браузер не може знайти файл таблиці стилів, заданий в інструкції по обробці xml-stylesheet, він відобразить текст документа з використанням свого власного набору властивостей (наприклад, з поточними значеннями гарнітури і розміру шрифту).

Можливо включити в XML-документ більш однієї таблиці стилів, вставивши для кожної з них інструкцію по обробці xml-stylesheet на початку XML- документа, наприклад:

<?xml version="1.0"?>

<?xml-stylesheet type="text/css" href="Book01.css"?>

<?xml-stylesheet type="text/css" href="Book02.css"?>

<INVENTORY>

<!—вміст елемента Документ -->

</INVENTORY>

Можливість зв'язування з декількома таблицями стилів дозволяє зберігати групи взаємопов'язаних правил в окремих файлах, а потім об'єднувати їх при створенні документів певних типів.

## Пріоритет правил каскадних таблиць стилів

У таблицях каскадних стилів можна присвоювати значення властивостям на декількох різних рівнях. Основні рівні, на яких можна привласнювати значення властивості – від вищого рівня пріоритету до нижчого.

Значення властивості в атрибуті STYLE для певного елемента в XML- документі, має найвищий пріоритет. Якщо не встановлено властивість в атрибуті STYLE, браузер використовує значення властивості, оголошеного в правилі CSS з контекстуальних селектором. Якщо не оголошено значення певної властивості в правилі, що має відповідний контекстуальний селектор, браузер використовує значення, оголошене в правилі з родовим селектором (тобто селектором, який включає тільки ім'я елемента). Якщо не оголошено значення для певного властивості для елементу в родовому правилі, браузер використовує установку властивості, оголошену для найближчого елемента-предка (батька, батька батька і т.д.).

Якщо таблиця стилів не містить установку властивості для будь-якого батьківського елемента, браузер використовує свою власну установку. Такий установкою може бути значення за замовчуванням, вбудоване в браузер, або значення, задане користувачем браузера. Наприклад, якщо не встановлено значення для властивості font-family, браузер використовує своє власне значення цієї властивості для відображення всіх елементів (В Internet Explorer це шрифт Times New Roman, якщо тільки користувач браузера не вибере інше сімейство шрифтів, скориставшись командою Internet Options – Властивості оглядача, з меню Tools (Сервіс).

Якщо для певного властивості встановлені конфліктуючі значення на одному і тому ж рівні, то в такому випадку браузер використовує останню установку, яку він обробив.

Цей процес має місце тільки для успадкованих властивостей. Для неуспадкованих властивостей браузер використовує значення властивості за замовчуванням.

## Властивості CSS, які часто використовуються в XML-документах

*Установка властивості* display

Властивість display керує основним способом відображення тексту елемента браузером. Можливо призначити йому три ключових слова CSS. Властивість display не успадковується дочірніми елементами. Можливі значення властивості:

Block – браузер завжди поміщає новий рядок перед і після тексту елемента (який включає і текст, що належить будь-яким дочірнім елементам). В результаті текст елемента відображається в окремому "блоці" з попереднім текстом вище і подальшим текстом нижче. Присвоєння значення block дозволяє форматувати текст із застосуванням різних властивостей обрамлення до блоку тексту, таких як поля, рамки і відступи. Елемент block схожий на абзац, який відділений пробілами від попереднього і наступного тексту, і для якого можна задавати відступи, рамки і т.д.;

inline (за замовчуванням) – браузер не вставляє новий рядок перед або після тексту елемента (якщо тільки попередній текст досяг правої межі вікна, і браузер повинен перенести текст на наступний рядок). Він буде вставляти перенесення рядка всередині тексту елемента тільки з метою вмістити текст у вікні. Текст елемента, таким чином, може бути розміщений в тому самому рядку, що і попередній або наступний текст. Елемент inline аналогічний групі символів всередині абзацу в програмі текстового процесора;

none – браузер не відображає елемент. Можливо використовувати цю установку для елементів, що несуть інформацію, яку не бажано поміщати на екрані.

## Установка властивостей шрифту

У стандартній CSS-таблиці передбачені наступні властивості, що визначають вид шрифту, який використовується для відображення тексту елемента:

font-family ; font-size ; font-style ; font-weight ; font-variant.

Всі ці властивості успадковуються дочірніми елементами.

## Установка властивостей фону

Стандарт CSS підтримує такі властивості, що дозволяють модифікувати фонове оформлення елемента:

background-color; background-image; background-repeat;

background-position.

Технічно дочірні елементи не успадковують властивостей фону. Однак за замовчуванням фон елемента є прозорим. Це означає, що якщо опустити всі властивості фону для дочірнього елемента, буде видно колір фону або малюнок батьківського елемента (або браузера), тобто фон дочірнього елемента буде таким же, що і фон батьківського елемента.

## Установка властивостей розбивки тексту і вирівнювання

Стандарт CSS підтримує такі властивості, що дозволяють модифікувати розбивку, вирівнювання та інші характеристики тексту:

letter-spacing ; vertical-align ; text-align ; text-indent ; line-height ; text-transform ; text-decoration.

Дочірні елементи успадковують всі ці властивості, за винятком vertical- align.

## Установка властивостей текстових областей

Специфікація CSS підтримує ряд властивостей (типу box), які можна використовувати для форматування блоку тексту, що належить елементу. До цих властивостей відносяться наступні:

* властивість margin додає невидиме (прозоре) поле навколо елемента, зовні від видимої рамки (якщо вона є);
* властивість border відображає видиму рамку певного стилю навколо елемента;
* властивість padding додає просвіт безпосередньо зовні від вмісту елемента, але всередині рамки (якщо вона є);
* властивості завдання розмірів height і width встановлюють розміри області вмісту елемента з доданими просвітом і рамкою;

властивості завдання позицій float і clear встановлюють положення елемента щодо оточуючих елементів.

Елемент block – це елемент, для властивості display з встановленим значенням block, а елемент inline – це елемент, для властивості display з встановленим значенням inline. В Internet Explorer перші три групи властивостей (margin, border і padding) діють тільки на елементи block. І inline.

## Установка властивостей управління полями

За замовчуванням ширина полів навколо елемента дорівнює нулю. Щоб додати поле з однією або з декількох сторін елемента, можна привласнити нульове значення одному або декільком з наступних властивостей: margin-top;

margin-right; margin-bottom; margin-left.

Можливо використовувати стенографічні властивості. Установка

властивостей управління обрамленням. Для створення видимого обрамлення навколо елемента можна скористатися наступними властивостями CSS:

border-style; border-width; border-color.

## Установка властивостей просвіту між обрамленням і текстом

Властивість просвіт (padding) додають простір безпосередньо навколо вмісту елементу, всередині від наявної навколо елемента рамки. Без просвіту кордону рамки розташовуються безпосередньо поблизу тексту елемента. Додавання просвіту покращує сприйняття рамки. За замовчуванням ширина просвіту для елемента встановлюється близькою до нуля. Щоб додати просвіт з одного або з декількох сторін від тексту елемента, можна привласнити нульове значення одному або декільком з наступних властивостей:

padding-top; padding-right; padding-bottom; padding-left.

Можливо встановлювати для цих властивостей значення в будь-яких розмірних одиницях, що допускаються в CSS.

Як і для вмісту елемента, для просвіту відображається будь-фон у вигляді суцільної заливки або малюнків, які призначені для елемента. На відміну від області полів, для яких відображається фон батьківського елемента.

# ПРИКЛАД ВИКОНАННЯ РОБОТИ

Опрацюйте нижче запропонований приклад створення XML-документа і його відображення з допомогою каскадних таблиць стилів CCS.

Створіть у текстовому редакторі Notepad новий файл і введіть текст XML- документа, зберігши з розширенням .xml.

<?xml version="1.0" encoding="utf-8"?>

<zavdannya>

<STUDENT>

<FAMILI>Сидоров</FAMILI>

<NAME>Иван</NAME>

<YEAR>2005</YEAR>

<GROUP>KH-19</GROUP>

</STUDENT>

<STUDENT>

<FAMILI>Иванова</FAMILI>

<NAME>Клавдия</NAME>

<YEAR>2003</YEAR>

<GROUP>KH-18</GROUP>

</STUDENT>

<STUDENT>

<FAMILI>Журавлева</FAMILI>

<NAME>Анастасия</NAME>

<YEAR>2004</YEAR>

<GROUP>KH-31</GROUP>

</STUDENT>

<STUDENT>

<FAMILI>Семенников</FAMILI>

<NAME>Антон</NAME>

<YEAR>2006</YEAR>

<GROUP>KH-20</GROUP>

</STUDENT>

</zavdannya>

Даний документ складається з двох основних частин: прологу і кореневого документа (званого також елементом документа). Елемент документа називається тут zavdannya, його початковий тег – <zavdannya>, а кінцевий – </zavdannya>, а вміст – чотири вкладених елемента STUDENT. У свою чергу кожен елемент STUDENT містить ряд вкладених елементів.

Відкрийте документ за допомогою браузера. Після перевірки синтаксису, документ відобразиться на екрані. При наявності помилок замість документа на екран буде видано повідомлення про неможливість відобразити сторінку.

Спробуйте змінити ступінь деталізації представлення елементів документа. Клацніть на символі знака мінус (-) зліва від початкового тега, щоб згорнути елемент, або на знаку плюс (+) поруч із згорнутим елементом, щоб розгорнути його.

Створіть у файлі file.css каскадну таблицю:

STUDENT

{display: block; margin-top: 12pt; font-size: 10pt}

FAMILY

{font-style: italic}

NAME

{font-weight: bold}

Відкрийте в текстовому редакторі файл, створений в першому пункті завдання, і другим рядком документа наступну інструкцію по обробці:

<?xml version="1.0" encoding="utf-8"?>

<?xml-stylesheet type=”text/css” href=”file.css”?>

<! -- Ім’я файлу: fale\_10.xml -->

<zavdannya>

<STUDENT>

**ВАРІАНТИ ЗАВДАНЬ:**

Використовуйте створений у лабораторній роботі №1 XML-документ, згідно з обраним варіантом завдання, що представляє інформацію по певній предметної області.

Створений XML-документ повинен відповідати наступним вимогам:

* документи повинні мати глибину вкладеності не менше чотирьох елементів;
* число елементів документа, які не мають вкладених, має бути не менше п'яти;
* елементи документа повинні містити коментарі про свій зміст;
* документ повинен включати елементи, що містять символьні дані і дочірній елементи.

Якщо необхідно доробити до вимог цей документ, – внесіть зміни.

Створіть таблицю каскадних стилів, яка отформатує створений XML- документ. Створена CSS-таблиця повинна відповідати наступним правилам:

* CSS-таблиця повинна включати як контекстуальні, так і родові селектори;
* дочірні елементи повинні наслідувати CSS-формат батьківського елемента;
* створена CSS-таблиця повинна імпортувати іншу таблицю стилів;
* таблиця стилів повинна включати використання атрибута STYLE;
* створена таблиця стилів повинна включати:
  + властивості шрифту;
  + властивість фону;
  + властивості розбивки і вирівнювання тексту;
  + властивості текстових областей;
  + властивості полів і кордонів;
  + псевдо елементи.

У звіті до виконання лабораторної роботи №2 необхідно представити код xml-документа, код таблиць CSS та скріншот табличного представлення документа.

# КОНТРОЛЬНІ ПИТАННЯ

1. Що являє собою XML-документ?
2. Для чого використовується XML?
3. З яких розділів складається XML-документ?
4. Яку інформацію містить декларація XML-документа?
5. Яка інформація розміщується в пролозі XML-документа?
6. Як будуються елементи XML-документа?
7. Які типи елементів ви знаєте?
8. Що таке атрибути?
9. Які синтаксичні правила створення атрибутів XML- документа?
10. Для чого в XML-документах використовуються коментарі?
11. Для чого в XML-документі необхідно простір імен?
12. Визначте мету застосування каскадних таблиць стилів CSS для відображення XML-документу.
13. Сформулюйте основні правила складання таблиць стилів.
14. Зазначте умови використання контекстуальних селекторів.
15. Зазначте умови використання родового селектора generic.
16. Переваги використання атрибуту STYLE у каскадних таблицях стилів.
17. Сформулюйте Пріоритет правил каскадних таблиць стилів.
18. Застосування механізму імпорту інших таблиць стилів.
19. Яким чином зв'язуються таблиці стилів з XML-документом?
20. Перелічите властивості CSS, які найчастіше використовуються в XML- документах.
21. Переваги та недоліки використання аскадних таблиць стилів CSS для відображення XML-документу.

ЛІТЕРАТУРА

Основна:

1. Гнатовська Г.А. XML-технології. Конспект лекцій. ОДЕКУ, 2019. – 75 с.

2. С.В. Одиночкина Основы технологий XML – СПб: НИУ ИТМО, 2013. – 56 с.

3. Проценко О. Б. Web-програмування та web-дизайн. Технологія XML.

Навчальний посібник. – Суми: Видавництво СумДУ, 2009. – 126 с.

Додаткова:

4. Кох Д., Дэвидсон К. XML. Огромные возможности и легкость изучения.

Самоучитель. Пер. с англ. Чайкина А. И. – М.: НТ Пресс, 2007. – 256 с.

5. К.А. Кулаков, В М. Димитров. Технологии XML:в 2 ч. Часть 1: Организация

данных. Учебное пособие для студентов математического факультета.

Петрозаводск : Изд-во ПетрГУ, 2014. – 68 c.

6. Рей Э. Изучаем XML. – СПб.: Символ-Плюс, 2001. – 408с.

7. Эдди С. XML.Справочник. – СПб.: Питер, 2000. – 480 с.

8. Холзнер С. Энциклопедия XML. – СПб.: Питер, 2004. – 1101 с.

9. Шеперд Д. Освой самостоятельно XML за 21 день. – СПб.: Вильямс, 2002. –

432 с.

10. Старигин А. XML. Разработка Web-приложений. – СПб.: BHV, 2003. – 585 с.