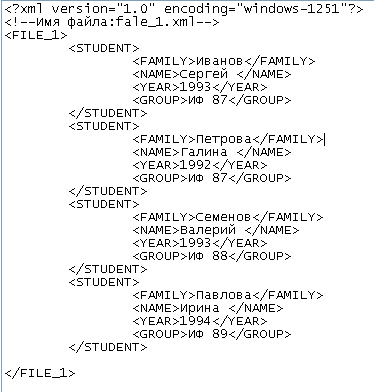
**Лабораторная работа № 10. Создание документа XML**

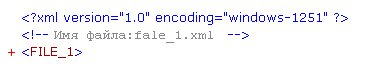
### Вопросы и задания к лабораторно работе № 1

1. Проработайте ниже предложенный пример создания XML-документа и его отображения с помощью каскадных таблиц стилей.
2. Создайте в текстовом редакторе Notepad новый файл и введите текст XML-документа, сохранив с расширением .xml

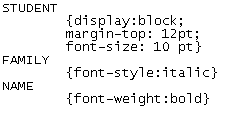


Данный документ состоит из двух основных частей: ***пролога*** и ***корневого документа*** (называемого также ***элементом документа***). Элемент документа называется здесь FILE\_1, его начальный тег - <FILE\_1>, а конечный - </FILE\_1>, а содержимое - 4 вложенных элементаSTUDENT. В свою очередь каждый элемент STUDENT содержит ряд вложенных элементов.

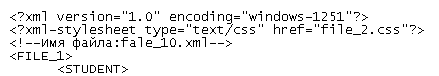
1. Откройте документ с помощью браузера Internet Explorer. После проверки синтаксиса, документ отобразится на экране. При наличии ошибок вместо документа на экран будет выдано сообщение о невозможности отобразить страницу.
2. Попробуйте изменить степень детализации представления элементов документа. Щелкните на символе знака минус (-) слева от начального тега, чтобы свернуть элемент, либо на знаке плюс (+) рядом со свернутым элементом, чтобы развернуть его. Например, щелкнув на знаке минус (-) рядом с элементом FILE\_1, вы получите то же, что представлено на рисунке:



1. Создайте в файле file\_2.css каскадную таблицу стилей:



1. Откройте в текстовом редакторе файл, созданный в первом пункте задания, и второй строкой документа следующую инструкцию по обработке:



1. Создайте XML- документ, представляющий информацию по определенной вариантом предметной области. Созданный документ должен соответствовать следующим требования:
   1. документы должны иметь глубину вложенности не менее четырех элементов;
   2. число элементов документа, не имеющих вложенных, должно быть не менее пяти;
   3. элементы документа должны содержать комментарии о своем содержании;
   4. документ должен включать элементы, содержащие символьные данные и дочерний элементы;
2. Создайте таблицу каскадных стилей, которая отформатирует созданный XML-документ. Созданная CSS-таблица должна соответствовать следующим правилам:
   1. CSS-таблица должна включать как контекстуальные, так и родовые селекторы;
   2. дочерние элементы должны наследовать CSS-формат родительского элемента;
   3. созданная CSS-таблица должна импортировать другую таблицу стилей;
   4. таблица стилей должна включать использование атрибута STYLE;

созданная таблица стилей должна включать:

* 1. свойства шрифта;
  2. свойство фона;
  3. свойства разбивки и выравнивания текста;
  4. свойства текстовых областей;
  5. свойства полей и границ;
  6. псевдоэлементы

1. В отчете представить код xml-документа, код таблиц CSSи скриншот табличного представления документа.

Варианты предметных областей создаваемых XML-документов:

|  |  |
| --- | --- |
| вариант 1: | библиографическое описание списка литературы |
| вариант 2: | описание фильмов видеотеки |
| вариант 3: | список сотрудников организации |
| вариант 4: | список моделей мобильных телефонов |
| вариант 5: | список студентов факультета |
| вариант 6: | список изучаемых дисциплин |

## Методический материал к лабораторной работе № 1

Вопросы и задания к лабораторно работе № 1 1

Методический материал к лабораторно работе № 1. 3

1.Введение в XML. 3

2.Отображение XML-документов с использованием каскадной таблицы стилей. 4

3.Связывание таблицы стилей с XML-документом 6

4.Приоритет правил каскадных таблиц стилей. 7

5.Cвойства CSS, часто используемые в XML - документах 8

### Введение в элементы XML

Описание XML- документа представляет собой простой текст, который можно набрать в любом текстовом редакторе. Создайте в текстовом редакторе Notepad новый файл и введите текст XML-документа, сохранив с расширением *.xml*

Каждый XML- документ размечается тегами.***Тег***

- это текст, заключенный в угловые скобки, который не относится к содержанию документа, а отмечает начало или конец какого-либо документа.

***Элемент*** документа XML имеет начальный тег, задающий имя элемента и дополнительную информацию (атрибуты), и конечный тег, содержащий то же имя элемента со знаком "/" впереди. Элементы определяют логическую структуру документа и несут в себе информацию, содержащуюся в документе. Имя элемента считается так же его типом.

Элементы XML-документа могут быть вложенными.

Пока не задан формат отображения XML-документа на экране, браузер применяет способ, принятый по умолчанию (IE будет использовать встроенную таблицу стилей для отражения документа). Одним из вариантов указания способа отображения документа является создание для него таблицы каскадных стилей (CSS).

**Правильно оформленными (well-formed) XML-документом** называется документ, удовлетворяющий минимальному набору правил соответствия для XML-документа.

Правильно оформленный XML-документ состоит из двух основных частей:***пролога*** и ***корневого элемента***. Помимо этого он может содержать***комментарии***,***инструкции***и***пробелы***. Корневой элемент может содержать вложенные элементы. Элементы должны быть правильно вложены. Если элемент начинается внутри некоторого другого элемента, то и заканчиваться он должен внутри того же элемента.

Каждый элемент состоит из начального тега, содержимого и конечного тега. Исключением является пустой элемент, для которого может использоваться единственный тег пустого элемента <Emptylement/>.

Правила использования имен элементов (***типов элемента***):

* имя должно начинаться с буквы или символа подчеркивания (\_),
* следующие после первого символы могут быть буквами, цифры, точкой, тире или подчеркиванием.
* Не следует использовать имена, начинающиеся с префикса "xml"(в любом сочетании строчных или прописных букв).
* Имя, записанное в начальном теге, должно в точности соответствовать имени в конечном теге;
* Соблюдение регистра существенно для имен элементов, как и для всего текста в описании разметки.

***Содержимым элемента***считается текст, расположенный между начальным и конечным тегами.

В начальный тег элемента либо в тег пустого элемента можно включить один или несколько описаний атрибутов. Описание атрибута представляет собой пару ***имя = значение***, связанную с данным элементом. Каждое имя атрибута может только один раз присутствовать в начальном теге элемента. Правила именования атрибутов аналогичны правилам именования элементов. Задание атрибутов обеспечивает альтернативный способ включения информации в элемент. Значение, которое можно присваивать атрибуту, представляет собой строку символов (литерал), ограниченных одинарными или двойными кавычками.

***Комментарий***начинается с символов**<!--**и заканчивается символами**-->.**Внутри комментариев не может содержаться двойное тире (--), символ левой угловой скобки (<) и знак амперсанда (&). Комментарии можно вставлять в любое место XML-документа, кроме описания тега.

***Инструкции***(инструкции по обработке) предназначены для XML-процессора, который будет обрабатывать документ. Общий вид инструкции:

***<?наименование данные?>***

***Наименование***указывает получателя инструкции. Параметр***данные***задает содержание инструкции. По наименованию конкретный XML-процессор определяет, предназначены данные ему или другому процессору. Инструкции могут помещаться в любое место XML-документа вне описания тегов.

Инструкция <?xml-stylesheet type="text/css" href="file\_2.css"?> предписывает Internet Explorer 5 использовать CSS-таблицу из файла file\_2.css.

Внутри символьных данных в содержимом элемента нельзя помещать некоторые специальные символы (<, &, :), так как это может привести к путанице при обработке документа. Одним из возможных путей преодоления этих ограничений является использование ***разделов символьных данных***. Такой раздел начинается с символов**<![CDATA[**и заканчивается**]]>**. Все символы внутри разделаCDATAрассматриваются как литеральная часть символьных данных элемента, а не как XML-разметка. РазделCDATAможет располагаться в любом месте документа, занимаемом символьными данными. РазделыCDATAне могут быть вложенными.

Документ XML может содержать ***пустые строки***, состоящие из одного или нескольких пробелов, символов табуляции, символаEnter. Можно свободно добавлять пробелы и переводы строк между:

* начальными и конечными тегами;
* комментариями;
* инструкциями по обработке.

Недолжно быть пробелов между открывающейся угловой скобкой и имеем элемента.

### Отображение XML-документов с использованием каскадной таблицы стилей.

Отображение XML-документов с использованием каскадной таблицы стилей осуществляется в два этапа:

1. создание файла таблицы стилей.
2. связывание таблицы стилей с XML-документом.

Таблица стилей состоит из одного или нескольких правил (иногда их называют набором правил). Правило содержит информацию по отображению определенного типа элемента в XML-документе.

Селектор представляет собой имя типа элемента, к которому относится информация по отображению.

Каскадные таблицы стилей CSS не чувствительны к регистру.

Набор наследуемых свойств, которые присвоены определенному элементу, действует на все дочерние элементы, прямо или косвенно вложенные в него, если только они не переустанавливаются впоследствии для определенного дочернего элемента.

Для ненаследуемых свойств, если не задано значение свойства для конкретного элемента, браузер использует значение свойства по умолчанию.

### 2.1. Использование контекстуальных селекторов

***Контекстуальным (contextual) селектором***называется селектор, в котором имя элемента может предваряться именами одного или нескольких элементов-предков (родительский, родительский плюс родительский родителя и т.д.). В этом случае правило будет применено только к элементам с этим именем, которые являются вложенными подобным образом. Между именами элементов в контекстном селекторе ставят только пробелы.

***Родовым селектором(generic)***называется селектор, который не включает имен элементов-предков.

Если определенное свойство для одного и того же элемента имеет одну установку в правиле с контекстуальным селектором, и другую установку в правиле с родовым селектором, установка в правиле с контекстуальным селектором доминирует, поскольку является более конкретизированной.

Например, фрагмент программного кода XML-документа, где <MAPS> - корневой элемент:

<MAPS>

<CITY>

<NAME>Santa Fe</NAME>

<STATE>New Mexico</STATE>

</CITY>

<STATE>California</STATE>

</MAPS>

Правила присоединенной таблицы стилей:

CITY STATE

{font-style:normal}

STATE

{font-style:italic}

В результате браузер отформатирует "New Mexico" обычным шрифтом, а "California" - курсивом.

### 2.2. Использование атрибута STYLE

Можно в XML-документе использовать специальный атрибут STYLE , вместо того, чтобы устанавливать одно или несколько определенных свойств отдельного элемента в таблице стилей.

Если значение свойства, установленного с помощью атрибута STYLE, конфликтует со значением свойства, установленного в таблице стилей, установка с помощью атрибута STYLE имеет приоритет. Таким образом, атрибут STYLE является удобным средством, чтобы переустановить - для определенного элемента - значение свойства, присвоенное для типа элемента в присоединенной таблице стилей.

Использование атрибута STYLE нарушает принцип CSS относительно хранения информации о форматировании отдельно от определения содержимого документа и структуры XML-файла.

Чтобы установить одно или несколько значений свойств, объявления в значение атрибутаSTYLEвключаетсяв виде строки, заключенной в кавычки, отделяя индивидуальные объявления точкой с запятой.

Например,

<TITLE STYLE='font-style:normal; font-size:14pt'>

The Adventures of Huckleberry Finn

</TITLE>

### 2.3. Импорт других таблиц стилей

Можно с помощью директивы ***@import***  в одну каскадную таблицу стилей встроить одну или несколько других таблиц стилей. Возможность импорта отдельных таблиц стилей позволяет хранить правила для связанных стилей в отдельных файлах, а затем объединять их при создании документов определенного типа. Директива @import должна располагаться в начале таблицы стилей перед правилами. В начале таблицы стилей можно разместить несколько директив @import.

***@import url(URL ИмпортТаблСтил);***

Например,

/\* Имя файла: Inventory01.css \*/

@import url(Book.css);

BOOK

{display:block;

margin-top:12pt;

font-size:10pt}

/\* продолжение таблицы стилей: \*/

В случае возникновения конфликта правил основная таблица стилей (из файла, в который осуществляется импорт) имеет приоритет над импортируемыми таблицами стилей. Если импортируеся несколько таблиц стилей, правила из таблицы стилей, импортированной последней, имеют приоритет над правилами из ранее импортированных таблиц стилей.

### Связывание таблицы стилей с XML-документом

Чтобы связать таблицу каскадных стилей с XML-документом, необходимо вставить в документ зарезервированную инструкцию по обработке ***xml-stylesheet***. Эта инструкция по обработке имеет следующую обобщенную форму записи, где URLТаблСтил есть URL, задающий местонахождение файла таблицы стилей:

**<?xml-stylesheet type="text/css" href="URLТаблСтил"?>**

Можно использовать полный URL, например:

**<?xml-stylesheet type="text/css"**

**href="http://www.my\_domain.com/Inventory01.css"?>**

Чаще используется частичный URL, который задает местонахождение относительно местонахождения XML-документа, содержащего инструкцию по обработке xml-stylesheet, например:

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

Обычно инструкция по обработке xml-stylesheet  добавляется в пролог XML-документа, вслед за объявлением XML. Если браузер не может найти файл таблицы стилей, заданный в инструкции по обработке xml-stylesheet, он отобразит текст документа с использованием своего собственного набора свойств (например, с текущими значениями гарнитуры и размера шрифта).

Можно включить в XML-документ более одной таблицы стилей, вставив для каждой из них инструкцию по обработке xml-stylesheet в начале XML-документа, например:

<?xml version="1.0"?>

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

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

<INVENTORY>

<!-- содержимое элемента Документ -->

</INVENTORY>

Возможность связывания с несколькими таблицами стилей позволяет вам хранить группы взаимосвязанных правил в отдельных файлах, а затем объединять их при создании документов определенных типов.

### Приоритет правил каскадных таблиц стилей.

В таблицах каскадных стилей можно присваивать значения свойствам на нескольких различных уровнях. Основные уровни, на которых можно присваивать значение свойству - от высшего уровня приоритета к низшему.

1. Значение свойства в атрибуте STYLE для определенного элемента в XML-документе, имеет наивысший приоритет.
2. Если не установлено свойство в атрибуте STYLE, браузер использует значение свойства, объявленного в правиле CSS с контекстуальным селектором.
3. Если не объявлено значение определенного свойства в правиле, имеющем соответствующий контекстуальный селектор, браузер использует значение, объявленное в правиле с родовым селектором (т.е. селектором, который включает только имя элемента).
4. Если не объявлено значение для определенного свойства для элемента в родовом правиле, браузер использует установку свойства, объявленную для ближайшего элемента-предка (родителя, родителя родителя и т.д.).
5. Если таблица стилей не содержит установку свойства для какого-либо родительского элемента, браузер использует свою собственную установку. Такой установкой может быть значение по умолчанию, встроенное в браузер, либо значение, заданное пользователем браузера. Например, если не установлено значение для свойства font-family, браузер использует свое собственное значение этого свойства для отображения всех элементов. [В Internet Explorer 5 это шрифт Times New Roman, если только пользователь браузера не выберет другое семейство шрифтов, воспользовавшись командой Internet Options (Свойства обозревателя) из меню Tools (Сервис)].
6. Если для определенного свойства установлены конфликтующие значения на одном и том же уровне, В таком случае браузер использует последнюю установку, которую он обработал.

Этот процесс имеет место только для наследуемых свойств. Для ненаследуемых свойств браузер использует значение свойства по умолчанию.

### 5. Cвойства CSS, часто используемые в XML - документах

### 5.1.Установка свойства display

***Свойство***display***управляет основным способом*** отображения текста элемента браузером. Вы можете назначить ему три ключевых слова CSS. Свойство display не наследуется дочерними элементами. Возможные значения свойства:

* **block**. Браузер всегда помещает перевод строки перед и после текста элемента (который включает и текст, принадлежащий любым дочерним элементам). В результате текст элемента отображается в отдельном "блоке" с предшествующим текстом выше и последующим текстом ниже. Присвоение значения block позволяет форматировать текст с применением различных свойств обрамления к блоку текста, таких как поля, рамки и отступы. Элемент block похож на абзац в программе текстового процессора, который отделен пробелами от предшествующего и последующего текста, и для которого можно задавать отступы, рамки и т.д.;
* **inline (по умолчанию)**. Браузер не вставляет перевод строки перед или после текста элемента (если только предшествующий текст достиг правой границы окна, и браузер должен перенести текст на следующую строку). Он будет вставлять перенос строки внутри текста элемента только с целью уместить текст в окне. Текст элемента, таким образом, может быть размещен в той же строке, что и предыдущий или последующий текст. Элемент inline аналогичен группе символов внутри абзаца в программе текстового процессора;
* **none**. Браузер не отображает элемент. Вы можете использовать эту установку для элементов, несущих информацию, которую вы не хотели бы помещать на экране.

### 5.2. Установка свойств шрифта

В стандартной CSS-таблице предусмотрены следующие свойства, определяющие вид шрифта, используемого для отображения текста элемента:

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

Все эти свойства наследуются дочерними элементами.

### 5.3. Установка свойств фона

Стандарт CSS поддерживает следующие свойства, позволяющие вам модифицировать фоновое оформление элемента:

* background-color;
* background-image;
* background-repeat;
* background-position.

Технически дочерние элементы не наследуют свойств фона. Однако по умолчанию фон элемента является прозрачным. Это означает, что если опустить все свойства фона для дочернего элемента, будут видны цвет фона или рисунок родительского элемента (или браузера), т.е. фон дочернего элемента будет таким же, что и фон родительского элемента.

### 5.4.Установка свойств разбивки текста и выравнивания

Стандарт CSS поддерживает следующие свойства, позволяющие модифицировать разбивку, выравнивание и другие характеристики текста:

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

Дочерние элементы наследуют все эти свойства, за исключением vertical-align.

### 5.4.Установка свойств текстовых областей

Спецификация CSS поддерживает ряд свойств (типа box ), которые можно использовать для форматирования блока текста, принадлежащего элементу. К этим свойствам относятся следующие:

* свойство ***margin*** добавляет невидимое (прозрачное) поле вокруг элемента, снаружи от видимой рамки (если она имеется);
* свойство ***border*** отображает видимую рамку - определенного стиля - вокруг элемента, снаружи от просвета (если он имеется);
* свойство ***padding*** добавляет просвет непосредственно снаружи от содержимого элемента, но внутри рамки (если она имеется);
* свойства задания размеров ***height*** и ***width*** устанавливают размеры области содержимого элемента с добавленными просветом и рамкой;
* свойства задания позиций ***float***и***clear***устанавливают положение элемента относительно окружающих элементов.

Элемент ***block*** - это элемент, для свойства ***display*** с установленным значением ***block***, а элемент ***inline*** - это элемент, для свойства ***display*** с установленным значением ***inline*** . В Internet Explorer 5.0 и 5.01 первые три группы свойств ( ***margin***,***border*** и ***padding*** ) действуют только на элементы ***block***. Однако, в Internet Explorer 5.5 можно использовать эти свойства как для элементов ***block***, так и для элементов ***inline***. При другой версии Internet Explorer (с 5.0 по 5.5) можно применять свойства позиционирования и для элементов ***block***, и для элементов ***inline*** ; однако наиболее эффективно и предсказуемо они работают с элементами ***block***..

### 5.5.Установка свойств управления полями

По умолчанию ширина полей вокруг элемента равна нулю. Чтобы добавить поле с одной или с нескольких сторон элемента, можно присвоить ненулевое значение одному или нескольким из следующих свойств:

* margin-top;
* margin-right;
* margin-bottom;
* margin-left.

Можно использовать стенографические свойства

### 5.6.Установка свойств управления обрамлением

Для создания видимого обрамления вокруг элемента можно воспользоваться следующими свойствами CSS:

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

### 5.6.Установка свойств просвета между обрамлением и текстом

Свойство просвет (padding) добавляют пространство непосредственно вокруг содержимого элемента, внутри от имеющейся вокруг элемента рамки. Без просвета границы рамки располагаются непосредственно вблизи текста элемента. Добавление просвета улучшает восприятие рамки.

По умолчанию ширина просвета для элемента устанавливается близкой к нулю. Чтобы добавить просвет с одной или с нескольких сторон от текста элемента, можно присвоить ненулевое значение одному или нескольким из следующих свойств:

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

Можно устанавливать для этих свойств значения в любых размерных единицах, допускаемых в CSS.

Как и для содержимого элемента, для просвета отображается любой фон в виде сплошной заливки или рисунков, которые назначены для элемента. (В отличие от области полей, для которых отображается фон родительского элемента.