МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Колледж экономики, управления и права

Методические рекомендации по организации самостоятельной работы студентов по ПМ.05 Разработка программных продуктов МДК.05.02 Современные веб-технологии

Таблица стилей CSS

Специальность

09.02.05 Прикладная информатика (по отраслям)

Ростов-на-Дону
2018

Методические рекомендации по ПМ.03 Разработка программных продуктов МДК.03.02 Современные веб-технологии разработаны с учетом ФГОС среднего профессионального образования специальности 09.02.05 Прикладная информатика (по отраслям), предназначены для студентов и преподавателей колледжа.

Методические рекомендации содержат цели изучения курса, задачи курса, методы обучения, планируемые результаты, содержание МДК, необходимый теоретический материал, команды для создания сайта, а также список рекомендуемой литературы.

Составитель (автор): С.В.Шинакова, преподаватель колледжа ЭУП

Рассмотрены на заседании предметной (цикловой) комиссии специальностей 09.02.04 Информационные системы (по отраслям) и 09.02.05 Прикладная информатика (по отраслям)

Протокол № 1 от «31» августа 2018 г

Председатель П(Ц)К специальности _____ С.В.Шинакова

и одобрены решением учебно-методического совета колледжа.

Протокол № 1 от «31» августа 2018 г

Председатель учебно-методического совета колледжа

_____ С.В.Шинакова

личная подпись

Рекомендованы к практическому применению в образовательном процессе.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

При создании web-страниц, основываясь на стандартных языках HTML (**H**yper**T**ext **M**arkup **L**anguage - язык разметки гипертекста) и XML (Extensible Markup Language – Расширяемый язык разметки), постепенно стали отказываться от многих встроенных дескрипторов управления стилем. Это объясняется распространением применения таблиц стилей CSS (Cascade Style Sheet) и расширяемого языка таблиц стилей XSL (Extensible Stylesheet Language), возможности которых гораздо шире.

Задача точного расположения объектов на web-странице относительно друг друга на компьютерах всех посетителей web-ресурса в независимости от настроек их браузеров является одной из наиболее сложных для web-авторов. Для решения этой и некоторых других задач оптимально подходят таблицы стилей, позволяющие переопределять свойства любого дескриптора, заданные по умолчанию, выравнивать блок текста относительно страницы или других блоков. Использование таблиц открывает специфические особенности, аналоги которых отсутствуют в статических языках разметки. Благодаря стилевым таблицам можно увеличить скорость создания страниц. Определив стиль один раз можно его использовать сколь угодно много раз в любом месте документа. Изменение свойств дескриптора в стилевом файле вызовет автоматическое изменение стиля отображения всех объектов, к которым применяется заданный стиль.

1 ОСНОВНЫЕ ПОНЯТИЯ И ПРИНЦИПЫ ИСПОЛЬЗОВАНИЯ ТАБЛИЦ CSS

CSS обычно используются в HTML-документах и широко поддерживаются различными браузерами. С помощью таблиц стилей можно определить форматирование индивидуальных элементов, создать классы стилей, установить шрифты, цвета, размещать элементы на странице и задавать другие правила отображение исходной информации в окне. XSL чаще используются при работе с XML. XSL – документы формируются в соответствии с правилами создания XML –документов. Таблица CSS определяет только формат и размещение элементов, а XSL может переупорядочивать элементы, полностью менять их, скрывать некоторые из них, выбирать стили основываясь как на этих элементах, так и на их атрибутах.

В связи с использованием в образовательном процессе принципа «от простого к сложному», в данных методических указаниях разберем основы применения таблиц CSS. Именно этот объект обеспечивает отображение документов в виде, изначально задуманном пользователем, в большинстве web-приложений. Существует два уровня таблиц CSS, причем для каждого имеется своя спецификация, рекомендованная к использованию консорциумом W3C. Уровень CSS2 включает все элементы, относящиеся к уровню CSS1, а также некоторые дополнительные свойства (таблицы стилей для звука, для различных

медиа-данных и т.д.). Уровень поддержки CSS различными браузерами различен. В Microsoft Internet Explorer, этот уровень, за исключением нюансов, достаточен для практически одинаковой работы. Хотя при начале работы по разработке web-приложения желательно проверить корректность интерпритации таблиц стилей Вашим браузером.

Таблица CSS представляет собой набор правил стилей, связанных с документом и определяющим форму его отображения web-браузером. При разработке таблицы стилей для каждого элемента документа может быть задано соответствующее правило. Правило состоит из селектора, определяющего элемент, на который будет направлено действие правила, и определения, заключенного в фигурные скобки, характеризующего особенность отображения документа. Определение в свою очередь состоит из параметра и его значения, отделенных двоеточием (рис. 1).

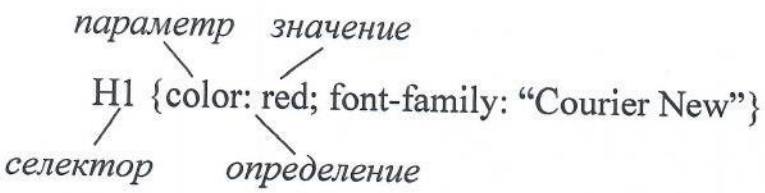

Рис. 1 Пример правила таблицы стилей

Правило, приведенное на рис. 1.1 задает отображение заголовка первого уровня (размера) красным цветом и типовым шрифтом “Courier New”. В одном правиле можно задать несколько определений, разделив их точкой с запятой (;), кавычки (“…”) используются если значение параметра состоит из нескольких частей разделенных пробелом.

1.1 Связывание таблиц стилей с web-документами

Для отображения документа согласно предписанным правилам CSS браузер должен знать о существовании и местоположении таблицы стилей. Чтобы удовлетворить этому условию, необходимо выполнить операцию связывания таблицы CSS и HTML-документа одним из четырех методов.

1) Связывание – позволяет использовать одну таблицу CSS, хранящуюся в отдельном файле, для форматирования нескольких страниц HTML. Причем выбранная таблица применяется ко всему документу. Связывание выполняется с помощью тэга <LINK> в разделе документа <HEAD>.

```
<HEAD>
<LINK REL=”stylesheet” TYPE=”text/css” HREF=”style1.css”>
</HEAD>
```

Параметр REL указывает тип присоединяемого к документу файла, параметр TYPE указывает тип стилевой таблицы, HREF обращается к местоположению таблицы. Использование данного метода позволяет применять одинаковый набор правил для группы документов для обеспечения их одинакового отображения в браузере.

2) Внедрение – задает все правила CSS в стилевом блоке самого документа, обеспечивая их применение ко всей странице. Стилевой блок находится в заголовочном разделе документа и ограничивается тэгами <STYLE> и символами комментариев для HTML, что позволяет исключить выполнение правил браузерами, не поддерживающими таблицы стилей.

```
<HEAD>
<STYLE TYPE="text/css">
<!--
H1 {color: red; font-family: "Courier New"; }
P {background-color: lightsteelblue; }
-->
</STYLE>
</HEAD>
```

3) Импортирование – позволяет связать с HTML-документом таблицу стилей, расположенную на сервере, и использовать ее для любого фрагмента. Внешняя таблица импортируется в тэге <STYLE> с помощью свойства *@import* таблиц стилей.

```
@import: url (style2.css);
```

Приведенное свойство указывается в начале стилевого блока или связываемой таблицы до блока с другими заданными правилами. В качестве значения этого свойства используется URL-ссылка, определяющая месторасположение стилевого файла.

4) Встраивание в дескрипторы документа – задает правила форматирования для отдельных элементов документа. Этот метод предназначен для адресного изменения свойств отображения определенного фрагмента страницы. В этом случае таблица стилей встраивается в отдельный тэг с помощью параметра STYLE, который может определяться практически для любого дескриптора. Значения этого параметра задаются в соответствии с синтаксисом каскадных таблиц стилей.

```
<H1 STYLE="color: red; font-size: 14pt"> Заголовок первого уровня отображается красным цветом, шрифтом размера 14 пунктов </H1>
```

Подобное использование таблиц стилей лишает разработчика преимуществ задания CSS в отдельном файле или в головной части документа, в случае необходимости изменения параметров отображения придется просмотреть весь документ. Этот метод похож на форматирование документа с помощью отдельных дескрипторов и их атрибутов.

1.2 Свойства группирования и наследования

Для уменьшения размеров таблиц стилей можно группировать разные правила по селектору или определению. Группирование по селектору используется при совпадении определений нескольких правил. Например, для таблицы

```
EM {font-style: oblique}
```

TT {font-style: oblique}
DFN {font-style: oblique}
можно выполнить группировку по селектору
EM, TT, DFN {font-style: oblique}

Смысл таблицы стилей от этого не меняется, она по прежнему задает отображение текста, размеченного тэгами выделения важных фрагментов, моноширинного шрифта и определений, наклонным стилем шрифта.

Группирование по определению используется при совпадении селекторов нескольких правил. Следующие правила

H1 {color: blue}
H1 {font-family: Arial}
H1 {letter-spacing: 2em}

можно записать сгруппировав определения:

H1 {color: blue;
font-family: Arial;
letter-spacing: 2em}

Некоторые свойства таблиц стилей имеют собственный синтаксис группирования и могут совмещать несколько значений разных параметров в одном правиле. Например, свойство *font* позволяет таблицу

U { font-weight: bold}
U {font-family: Arial}
U {font-size: 12pt}
U {font-style: normal}

представить в виде

U { font: bold Arial 12pt normal}

Все три правила группирования могут сочетаться друг с другом произвольным образом для уменьшения стилевой таблицы.

Наследование правил форматирования для HTML-документов основано на делении всех элементов на элементы–родители и элементы–потомки. Если для элемента-родителя задано правило форматирования, а для вложенного элемента нет, то он наследует свойства родителя. Например, если цвет шрифта абзаца определен как красный (P {color: red}), то текст в этом абзаце выделенный как полужирный (<B>), при отсутствии правил форматирования для селектора *B*, будет отображаться красным. Не все свойства родительских элементов наследуются вложенными элементами, например, это касается свойства *background* дескриптора <BODY>, хотя по умолчанию все вложенные элементы отображаются с фоном элемента-родителя.

Правила наследования могут применяться при задании значений атрибутов, применяемых к документам по умолчанию. Нужно задать все свойства для элементов-родителей, чтобы управлять свойствами элементов, порождаемыми ими. Для задания свойств для всего тела документа используется тэг <BODY>.

Задание

1. Составьте таблицу стилей из пяти приведенных правил и свяжите ее с HTML-документом каждым из четырех способов.

2. Выполните примеры, отображающие свойства группирования и наследования, описанные в данном параграфе.

2 ОСОБЕННОСТИ ИСПОЛЬЗОВАНИЯ РАЗЛИЧНЫХ СЕЛЕКТОРОВ

2.1 Специальные селекторы

Использование в качестве селектора HTML-дескриптора, изменяет форматирование внешнего вида всех элементов, относящихся к данной категории. Но часто требуется определить стиль форматирования только для некоторых элементов, объединенных в класс по какому-то признаку или не связанных между собой. Для этого применяются специальные селекторы к которым относятся селекторы CLASS, ID и контекстные селекторы.

Селектор CLASS

Этот селектор позволяет задавать правила форматирования для элемента, определенного конкретным дескриптором, или для всех элементов документа. Имя класса указывается в селекторе правила и отделяется от дескриптора точкой. Для одного тэга в таблице можно определять несколько стилей форматирования и с помощью параметра *CLASS* применять их в документе.

В таблице стилей правила с использованием классов записываются следующим образом:

```
<STYLE TYPE="text/css">
<!--
H1.red {color: red; }
H1.bgcol {background-color: grey; }
P.blue {color: blue; }
-->
</STYLE>
```

В тексте документа ссылка на элемент стилевой таблицы задается через имя класса, стоящее в селекторе после точки, которое указывается как значение параметра *CLASS*.

```
<H1 CLASS="red">Текст заголовка первого уровня отображается красным цветом </H1>
<H1 CLASS="bgcol">Текст заголовка первого уровня отображается на сером фоне</H1>
<H1>Текст отображается с параметрами установленными по умолчанию</H1>
<P CLASS="blue">Текст данного абзаца отображается голубым цветом</H1>
```

Таким образом, задаются классы для отображения элементов конкретного типа. Если правила форматирования следует применять к разным элементам одного документа, то селектор задается без указания дескриптора с именем класса и предшествующей точкой, указывающей на наличие объекта *CLASS*:

```
<STYLE TYPE="text/css">
<!--
.red {color: red; }
.bgcol {background-color: grey; }
.blue {color: blue; }
-->
</STYLE>
```

Свойства классов red, bgcol, blue можно применять к любым элементам документа:

```
<P CLASS="red">Текст данного абзаца отображается красным цветом </P>
<H3 CLASS="bgcol">Текст заголовка тертьего уровня отображается на сером фоне</H3>
<B>Текст отображается с параметрами установленными по умолчанию</B>
<EM CLASS="blue">Текст, выделенный курсивом, отображается голубым цветом</EM>
```

Селектор ID

В качестве специального селектора может применяться и селектор *ID*, определяющий уникальное название элемента таблицы стилей, указываемое после префикса (#). С его помощью устанавливаются ссылки в сценариях и в теле документа. Параметр *ID*, в качестве значения которого используется имя правила таблицы, может использоваться в любом дескрипторе документа.

В таблице стилей правила с использованием *ID* записываются следующим образом:

```
<STYLE TYPE="text/css">
<!--
#red {color: red; }
#bgcol {background-color: grey; }
-->
</STYLE>
```

В тексте документа ссылка на элемент стилевой таблицы задается следующим образом:

```
<H1 ID="red">Текст заголовка первого уровня отображается красным цветом </H1>
<TT ID="red">Моноширинный текст отображается красным цветом </TT>
<H1 ID="bgcol">Текст заголовка первого уровня отображается на сером фоне</H1>
<P ID="bgcol">В абзаце используется фон серого цвета</P>
```

Для форматирования конкретных элементов не следует злоупотреблять использованием идентификаторов отдельных частей документа. Лучше использовать правила форматирования, встраиваемые в тэги. Это связано с принципом приоритетности таблиц.

Контекстные селекторы

При создании HTML-документов может возникнуть задача форматирования вложенных элементов. Например, требуется изменить стиль отображения текста выделенного тэгом <DFN> во фрагменте документа, образованным тэгом <DIV>. Некоторые свойства элемента-родителя наследуются элементом-потомком, например, цвет шрифта. Для отображения вложенных элементов со своими свойствами их можно задать отдельно в стилевой таблице:

```
DIV {color: yellow}
DFN {color: silver}
```

В таком случае текст, отмеченный тэгом <DFN>, во всем документе будет отображаться шрифтом серебряного цвета. Если необходимо изменить стиль определения только в данном фрагменте текста используются контекстные селекторы, состоящие из нескольких дескрипторов отделенных пробелом.

```
DIV DFN {color: silver}
```

Интерпретатор браузера, просматривая документ, выявляет определения *DFN*, порожденные элементом *DIV*, и применяет к ним указанное правило форматирования. Правила с контекстными селекторами задаются, используя синтаксис таблиц стилей для одиночных селекторов.

2.2 Псевдоклассы используемые в HTML-документах

Правила форматирования применяются к элементам документа, имеющим определенное положение в структуре документа и использующим заданные атрибуты. Элементы могут объединяться в классы по расположению в структуре и по различным критериям. Если в классе можно выделить какую-то общность элементов, то она будет называться псевдоклассом, а элементы псевдоэлементами. Использование псевдоклассов и псевдоэлементов расширяет возможности адресации стилевых правил и позволяет внешней информации влиять на форматирование фрагментов документа. Псевдоэлементы используются для указания местоположения частей документа, а псевдоклассы – для форматирования стилей элементов различных типов.

В большинстве современных браузеров реализуется наиболее распространенный, а иногда и единственный, псевдокласс для элементов связей *A*.

Обычно браузеры отображают посещенные связи отлично от непосещенных, чаще всего используется цветовое отображение или отображение подчеркиванием.

Первый уровень CSS регламентирует отображение связей через псевдокласс элемента *A*:

```
A: link {color: aqua} /* непосещенная связь */
```

A: visited {color: blue} /* посещенная связь */

A: active {color: fuchsia} /* активная связь */

A: hover {color: maroon} /* связь, на которой расположен курсор манипулятора */

В каскадных таблицах символ /*...*/ используется для написания комментариев. Активной связью является связь, выбранная в данный момент, или связь в которой выполняется какой-то процесс, например, загружение или сохранение какой-то информации.

В данном примере связи отображаются различным цветом в зависимости от действий, совершенных пользователем.

Браузер ***Microsoft Internet Explorer 4.0*** правильно реализует указанные псевдоклассы связей, тогда как ***Netscape Navigator 4.0*** воспринимает только псевдоклассы *link* и *visited*. Поскольку псевдоклассы применяются только к типу элементов *A*, то справедлива запись правил без указания тэга в селекторе.

: link {color: aqua} /* непосещенная связь */

: visited {color: blue} /* посещенная связь */

2.3 Правила приоритетности при использовании таблиц стилей

По отношению к одному HTML-документу и отдельным его элементам могут применяться несколько стилевых таблиц, влияющих на его отображение в окне браузера. Правила этих таблиц могут задавать различные свойства отображения текста, но могут задавать одно и то же свойство с различными значениями. Для избежания несоответствия и конфликтов используется принцип приоритетности, когда правила с различными значениями одних и тех же свойств применяются последовательно. Таблицы стилей выстраиваются в каскад, в соответствии с которым происходит их применение в документе. Этот принцип является основополагающим в использовании CSS, он выполняет две функции:

1) Авторы web-страниц могут комбинировать стилевые таблицы, каждая из которых отвечает за форматирование какого-то этапа при отображении документа. Например, в одной таблице можно задать правила цветового отображения документа, во второй – правила форматирования шрифтов, в третьей правила позиционирования элементов, относительно друг друга и т.д.

2) Соблюдается равновесие между параметрами отображения, заданными разработчиком и приоритетами пользователя. И тот и другой может влиять на представление страницы через таблицу стилей. Браузер на основании принципа приоритетности будет использовать правила из таблиц автора или пользователя.

Правила определенные в таблице как имеющие больший приоритет обладают большим *весом*. Правила автора при прочих равных условиях имеют больший вес, чем правила пользователя. Правила пользователя и разработчика приоритетней, чем установленные по умолчанию настройки браузера.

В случае импортирования таблиц в документ извне приоритетность зависит от порядка их импортирования в страницу. Стилевая таблица позволяет использовать несколько операторов *@import*. Каждая последующая импортированная таблица задает правила с большим весом, чем предыдущая. Импортированная таблица имеет большую приоритетность, чем исходная таблица. Правила, задаваемые в самом документе, имеют больший вес, чем правила импортируемых таблиц стилей. Обычно таблицы импортируются до определения правил. Для увеличения веса правила можно использовать параметр *important*, задаваемый после определения, вес которого нужно увеличить.

P {background-color: blue ! important}

Правило пользователя со значением *important* имеет больший вес чем аналогичное правило разработчика, заданное без этого параметра.

Для преодоления различных конфликтов в процессе применения правил определяется значение приоритетности комбинации элемент-свойство по встроенному в браузер алгоритму:

1) Находятся все определения, применяемые к рассматриваемому элементу, из связанных с документом таблиц стилей. Применение определений происходит если соответствующие им селекторы соответствуют дескриптору элемента. Если соответствия нет, то используются наследственные свойства. При отсутствии наследуемого значения используется заданное по умолчанию.

2) Определения сортируются по явно заданным весам. Разделяются свойства с параметром *important* и без него.

3) Определения сортируются по авторству. Самый больший вес имеют таблицы разработчика, а таблицы пользователя приоритетней таблиц стилей, заданных по умолчанию для браузера.

4) Определения сортируются по числу специфичности селектора. Селектор с большим числом специфичности имеет больший вес, чем селектор с меньшим числом. Число специфичности обозначается *abc*, *a* – число селекторов *ID*, *b* – число селекторов *CLASS*, *c* – число дескрипторов, входящих в селектор. Приведем несколько пример определения числа специфичности:

EM {…} /* a=0 b=0 c=1, число специфичности =1 */

OL LI UL {…} /* a=0 b=0 c=3, число специфичности =3 */

H1.red {…} /* a=0 b=1 c=1, число специфичности =11 */

DIV P H1.red {…} /* a=0 b=1 c=3, число специфичности =13 */

#form {…} /* a=1 b=0 c=0, число специфичности =100 */

Псевдоклассы связей считаются как обычные классы.

5) Определения сортируются по порядку их размещения в таблице стилей. Заданное позже правило имеет приоритет.

Задание

1. Примените правила форматирования, включающие специальные селекторы, для текстовых элементов, размеченных в документе дескрипторами <B>, <EM>, <TD>, <DIV>.

2. Для одного и того же тэга HTML-документа запишите три правила, каждое из которых имеет специальный селектор и различные значения одного и того же свойства, и присоедините их с помощью метода связывания и внедрения. Определить число приоритетности для каждого правила.

3. С помощью псевдокласса связей задайте различные значения цвета указателя ссылки, в зависимости от действий пользователя.

3 МОДЕЛЬ ФОРМАТИРОВАНИЯ ЭЛЕМЕНТОВ

Для форматирования элементов в стилевых таблицах применяется модель, представляющая собой вложенные прямоугольники, в центре которых находится содержимое самого элемента (рис.2).

Блок содержимого элемента отделен от границы отступом. Внешней оболочкой является поле. Поле является прозрачным, его цвет наследует элемент-родитель (для текста это элемент *BODY*). Отступ имеет цвет фона самого элемента. Цвет границы задается отдельно. По особенностям форматирования различаются блочные и встроенные элементы.

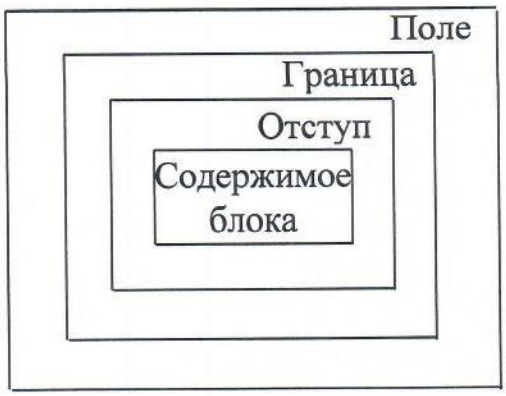

Рисунок 2 - Модель форматирования тестового блока

Блочные элементы

В модели на рис. 2.1 к каждому элементу может быть применено свойство *display* задающее отображение элемента со значениями *yes* или *no*, а также его вид блочный (block), встроенный (inline) или список (list-item). Элементы со значением свойства *display* равным *block*, *list-item*, а также элементы со значением параметра *float* равным *none* являются блочными элементами.

Для форматирования блочного элемента по горизонтали используются семь свойств: левое поле (margin-left), левая граница (border-left), левый отступ (padding-left), ширина элемента (width), правый отступ (padding-right), правая граница (border-right), правое поле (margin-right). Сумма значений параметров горизонтального выравнивания равна ширине элемента-родителя или ширине окна браузера, если отсутствует вложение.

Встроенные элементы

Если элементы не форматируются по правилам блочных элементов, то они являются встроенными элементами. Эти элементы для отображения используют область строки. В случае недостаточной ширины строки для отображения, элемент будет разбит на два, помещенных в разные строки. Приведем один пример встроенного элемента.

<P> В абзаце располагается

<EM STYLE="color:lime"> встроенный элемент, отображаемый красным цветом и курсивом

</EM> </P>

Задание

Задайте HTML-таблицу, поместите в одну ячейку изображение, а в другую текстовый фрагмент. Используя параметры тэгов <TABLE>, <TD> и правила таблиц стилей отформатируйте изображение таким образом, чтобы расстояние между границей и блоком было 20мм, а расстояние между блоками 35мм.

4 СВОЙСТВА, ВЛИЯЮЩИЕ НА ФОРМАТИРОВАНИЕ ЭЛЕМЕНТОВ

Все свойства форматирования HTML-документов, доступные в каскадных таблицах стилей первого и второго уровня, можно разделить на 9 категорий:

- шрифт: различные типографские свойства;
- текст: выравнивание, межсимвольное расстояние и форматирование;
- цвет: цвет текста;
- фон: цвет фона и фоновые изображения;
- блок: свойства форматирования отдельных блоков;
- визуальное форматирование: свойства, создающие блоки из отображаемых элементов и позиционирующие их, а также вывод списков;
- переходы и фильтры: определение мультимедийных эффектов, преобразование изображений;
- печать: свойства разметки страниц;
- псевдоклассы и другие свойства: свойства псевдокласса связей, свойства cursor, impotant и @import.

Для задания значений свойств таблиц стилей, определяющих размеры элементов, используются относительные и абсолютные единицы. Относительные единицы задают размер элемента относительно ранее заданной величины. Документы, использующие относительные единицы измерения, лучше приспособлены для отображения на различных устройствах. Абсолютные единицы желательно использовать в случаях, когда известны метрические характеристики устройства отображения.

Основные единицы измерения, используемые в web-дизайне, представим в табличном виде (Таблица 1).

Пикселы соотносятся с разрешением монитора персонального компьютера. Если плотность пикселов на единицу площади на устройстве вывода значительно отличается от значения для монитора, браузер переопределяет эту единицу. Таким образом, появляется новая единица, ссылочный или условный пиксел, которым является угол, под которым виден один пиксел монитора с плотностью 90 dpi с расстояния вытянутой руки (примерно 28 дюймов).

Таблица 1

Единицы измерения			
Относительные		Абсолютные	
em	Высота шрифта элемента	in	Дюйм (1 in=2.54 cm)
ex	Высота буквы	cm	Сантиметр
px	Пиксел	mm	Миллиметр
%	Процент	pt	Пункт (1 pt=1/72 in)
		pc	Пика (1 pc=12 pt)

4.1. Шрифты

Подбор шрифта документа, является одним из способов улучшения читаемости текста, выделения отдельных фрагментов и компоновки всего информативного объема. Шрифт может отличаться стилем и способом начертания, гарнитурой и капителью.

Свойство *font-family*

С помощью этого свойства определяется шрифт отображаемого текста или приоритетный перечень семейств шрифтов. Используя это свойство, разработчик может использовать перечень шрифтов, имеющих сходные характеристики (гарнитура и капитель), из которых браузер выбирает наиболее подходящий, в случае невозможности применить стандартный шрифт.

BODY {font-family: “Times New Roman Bold”, “Times New Roman”, serif}

При отображении страницы браузер сначала ищет шрифт *Times New Roman Bold*, если этот шрифт отсутствует, то ищется *Times New Roman*, если же и он не найден, то используется любой шрифт семейства *serif*.

При отсутствии требуемого шрифта используется одно из пяти наиболее распространенных семейств шрифтов:

- serif (шрифт с засечками, *Times New Roman*);
- sans-serif (шрифт без засечек, Helvetica);
- cursive (например, Arial);
- fantasy (например, Century Gothic);
- monospace (например, Courier New).

Свойство *font-style*

Это свойство определяет стиль выбранного шрифта: normal (обычный), italic (курсивный), oblique (наклонный). Наклонный шрифт формируется на основе обычного путем наклона вправо примерно на 15°, курсивный имеет ряд особенностей среди которых закругление уголков букв, что придает ему схожесть с рукописным шрифтом.

H1 {font-style: oblique}

Свойство *font-variant*

В стилевых таблицах реализовано еще одно свойство отображения шрифта выбранного семейства – капитель. В этом стиле все строчные буквы выглядят как прописные, но меньшего размера и с измененными пропорциями. Значение *normal* не изменяет вид шрифта, а значение small-caps выбирает капитель шрифта. Таблицы стилей допускают создание капители заменой строчных букв масштабированными символами верхнего регистра.

H4 {font-variant: small-caps}

Свойство *font-weight*

Это свойство позволяет задать степень толщины (жирности) шрифта. Существует 9 значений этого свойства, которые находятся в интервале от 100 до 900, и чередуются с шагом 100.

I {font-weight: 600}

Кроме числовых значений можно использовать и ключевые слова:

- normal: эквивалент числового значения 400;
- bold: соответствует числовому значению 700.

Свойство *font-size*

Данное свойство задает размер шрифта, при этом можно использовать как абсолютные, так и относительные единицы измерения. При использовании абсолютных единиц значение высоты шрифта будет фиксировано и не связано с таблицей размеров шрифтов данного браузера. Если использовать относительные единицы размер будет определяться с помощью процентных соотношений.

Для задания размера шрифта могут использоваться ключевые слова, отражающие как абсолютные, так и относительные величины. Абсолютный размер шрифта можно задать следующими словами, по которым браузер выбирает размер шрифта из соответствующих таблиц:

- xx-small;
- s-small;
- small;
- medium;
- large;
- x-large;
- xx-large.

Каждое следующее ключевое слово увеличивает размер шрифта, заданный предыдущим ключевым словом на 20%.

Относительные единицы измерения задаются ключевыми словами *larger* и *smaller*, которые сдвигают на один номер в соответствующую сторону размер шрифта, заданный абсолютными единицами.

```
P {font-size: 14pt}
H1 {font-size: 130%}
B {font-size: 1.3 em}
SPAN {font-size: medium}
DFN { font-size: larger}
```

Правила для селектора *H1* и *B* определяют одинаковую высоту шрифта.

Свойство *line-height*

Свойство задающее высоту строки, в качестве единиц измерения могут использоваться абсолютные и относительные единицы представленные в таблице 1.

```
P {font-size: 10pt}
H1 {line-height: 130%}
```

Свойство *font*

Поскольку это свойство обладает собственным синтаксисом группирования в нем можно задать значения перечисленных свойств: *font-family, font-style, font-variant, font-weight, font-size, line-height.* Если не определять значения второго, третьего и четвертого свойства, то они по умолчанию будут равны *normal*.

```
P {font: oblique 10pt/15pt bold "Times New", serif}
```

Размер шрифта (10pt) и высота строки (15pt) задаются через косую черту, свойство, определяющее семейство шрифтов, задается последним, т.к. шрифты перечисляются через запятую, в то время как все остальные значения задаются через пробел.

Свойство *@font-face*

Это свойство используется для загрузки шрифта с удаленного местоположения, определяемого URL-ссылкой, в случае если шрифт не установлен на локальном компьютере.

```
@font-face {font-family: Garamont;
src:url (http://homepage.org/Garamont.eot);}
```

4.2 Свойства, форматирующие текст

Данные свойства влияют на отображение символов, отдельных слов и текстовых параграфов. Может определяться промежуток между отдельными буквами и словами, высота строки параграфа и величины отдельных отступов. Для форматирования текста используются следующие свойства.

- text-transform: преобразовывает символы текста, принимает значения capitalize (все первые буквы слов преобразуются в заглавные), uppercase (все символы преобразуются в прописные буквы), lowercase (преобразование букв в символы нижнего регистра);

- text-decoration: элемент дополнительного оформления, отображает текст с горизонтальной чертой, overline (линия вверху текста), underline (подчеркивание), line-through (перечеркнутый текст);

- text-align: выравнивание текста, left (выравнивание по левому краю), center (выравнивание по центру), right (выравнивание по правому краю);

- text-indent: отступ текста внутри блока, задаваемый в абсолютных или относительных единицах измерения;

- vertical-align: вертикальное расположение элемента относительно элемента-родителя, baseline (выравнивание базовой линии элемента по базовой линии предка), middle (выравнивание средней линии элемента по базовой линии предка), sub (отображение элемента в регистре нижнего индекса), super (отображение элемента в регистре верхнего индекса), text-top (выравнивание верха элемента по верху шрифта элемента-родителя), text-bottom (выравнивание низа элемента по низу шрифта элемента-родителя);

- line-height: расстояние между базовыми линиями двух соседних строк.

4.3 Свойства, задающие цвет и фон

Набор свойств этой группы предназначен для определения или изменения цвета и фона, HTML-документа. Фон можно задать однородным или в виде изображения.

В случае задания цвета используется свойство *color*. Этому свойству присваивается значение цвета, определяемое с помощью стандартного имени, шестнадцатеричного кода или десятичного кода модели *rgb*:

```
H1 {color: red}
INS {color: #FF0000}
EM {color: rgb (255, 0, 0)}
```

В трех случаях для элементов задается красный цвет.

Для отображения фона можно использовать несколько параметров или свойство *background*. В случае задания фона однородным цветом используется свойство *background-color*, в качестве начального значения этого свойства используется transparent – прозрачный фон. Для задания в качестве фона какого-либо изображения применяется свойство *background-image*, значением которого является абсолютный или относительный адрес изображения.

Для задания повторяемости фонового изображения используется свойство *background-repeat* имеющее значение *repeat* (повторяемость в обоих направлениях), *repeat-x* (повторяемость по горизонтали), *repeat-y* (повторяемость по вертикали), *no-repeat* (отсутствие повторяемости).

Свойство *background-attachment* определяет прокрутку изображения вместе с текстом (scroll) или неподвижность изображения фона (fixed).

Свойство *background-position* задает начальное положение относительно блока содержащего элемент. В качестве значений могут применяться абсолютные, относительные величины и комбинации ключевых слов: top, top left, top right, center, center left, center right, bottom, bottom left, bottom right. Ключевые слова позволяют выравнивать фон с точностью 50%. Начало координат задается в левом верхнем углу блока.

BODY { *background-color*: DodgerBlue;
background-image: url(picture.gif);
background-repeat: repeat;
background-attachment: fixed;
background-position: bottom center}

4.4 Свойства форматирования отдельных блоков

К данному разделу относятся свойства для форматирования полей, обрамления и отступов блока. Данную группу свойств можно разделить на несколько подгрупп.

1). Свойства форматирования поля имеют числовое значение, отображающее ширину поля элемента:

- margin-top: верхнее поле элемента;
- margin-bottom: нижнее поле элемента;
- margin-left: левое поле элемента;
- margin-right: правое поле элемента;
- margin: устанавливает значение всех полей одновременно.

2). Свойства форматирования отступов имеют числовое значение, отображающее ширину отступа элемента:

- padding-top: верхний отступ элемента;
- padding-bottom: нижний отступ элемента;
- padding-left: левый отступ элемента;
- padding-right: правый отступ элемента;
- padding: устанавливает значение всех отступов одновременно.

3). Свойства форматирования обрамления, имеющие значение thin (тонкий), medium (средний), thick (толстый), устанавливают ширину обрамления элемента:

- border-top-width: верхнее обрамление элемента;
- border-bottom-width: нижнее обрамление элемента;
- border-left-width: левое обрамление элемента;
- border-right-width: правое обрамление элемента;
- border-width: устанавливает значение для толщины всех обрамлений одновременно.

4). Свойства, определяющие цвет обрамления:

- border-top-color: цвет верхнего обрамления элемента;
- border-bottom- color: цвет нижнего обрамления элемента;
- border-left- color: цвет левого обрамления элемента;

- border-right- color: цвет правого обрамления элемента;
- border- color: цвет обрамления со всех сторон элемента.

5). Свойства, определяющие тип обрамления:

- border-top-style: тип обрамления элемента сверху;
- border-bottom- style: тип обрамления элемента снизу;
- border-left-style: тип обрамления элемента слева;
- border-right-style: тип обрамления элемента справа;
- border-style: тип обрамления элемента.

Перечисленные свойства могут принимать следующие значения: none (обрамление отсутствует), solid (обрамление сплошной линией), double (обрамление двойной линией, сумма толщин линий и расстояние между ними устанавливается с помощью свойства border-width), groove (обрамление линией с углублением), ridge (обрамление линией с тенью), inset (элемент как будто находится ниже уровня фона), outset (элемент отображается как выпуклый).

Установка ширины, типа и цвета обрамления можно задать используя свойства: border-top, border-bottom, border-left, border-right, border.

4.5 Визуальное форматирование

Каскадные таблицы стилей являются эффективным средством, выполняющим компоновку элементов в окне браузера. Позиционирование HTML-элемента на странице выполняется с помощью свойства *position*, значением которого могут быть: absolute (абсолютный способ позиционирования), relative (относительный способ позиционирования), static (статический способ позиционирования).

Задание

С помощью тэгов <TABLE>, <TR>, <TD>, <TH> задайте таблицу с тремя колонками, в которую поместите список студентов своей подгруппы. Для отображения имен и фамилий используйте все свойства форматирования шрифта и текста, для задания цвета шрифта, фона таблицы и документа примените соответствующие свойства стилевых таблиц. Выравнивание выполните, используя свойства форматирования отдельных блоков.

5 ПРАКТИЧЕСКИЙ ПРИМЕР ИСПОЛЬЗОВАНИЯ КАСКАДНЫХ ТАБЛИЦ СТИЛЕЙ

Создавать блоки элементов для последующего их форматирования по различным правилам можно и без фреймовой структуры или обычных таблиц. Этому способствует реализация блочной модели в браузере. Каждый блок задается как отдельный слой, в который можно вкладывать другие слой, размеры и свойства форматирования для которых определяются отдельно. Отдельные проблемы могут возникнуть при установке точных размеров ячеек (блоков) и

при использовании для просмотра различных браузеров, но решение этих проблем лежит в совершенствовании умения разработчика.

Рассмотрим пример задания 7 блоков в одном документе, используя CSS и один дескриптор <DIV>:

```
<html>
<head>
<style type=text/css>
<!--
body {margin: 10px 10px 10px 10px;
      padding: 0;
      background: transperent;}
#level1 {border-style: solid;}
#level2 {margin-left: 150px;
padding-left: 10px;
border-style: double;
background: transperent;}
#level3 {border-style: groove;}
#level4 {margin-right: 150px;
      padding-right: 10px;
      border-style: double;
      background: transperent;}
#lewel5 {border-style: ridge;}
# lewel6 {background: blue;}
# lewel7 {background: steelblue;}
</style>
</head>
<body>
<div id=level1>
      <div id=level2>
            <div id= lewel6>
             Верхний блок
                  <div id= lewel6>
                  Средний блок
                  </div>
            </div>
       <div id=level3>
            <div id=level4>
                  <div id= lewel5>
                  Нижний вложенный блок
                  </div>
            </div>
       </div>
 </div>
 </div>
```

```
</body>
</html>
```

Результат данного кода можно увидеть на рис. 3

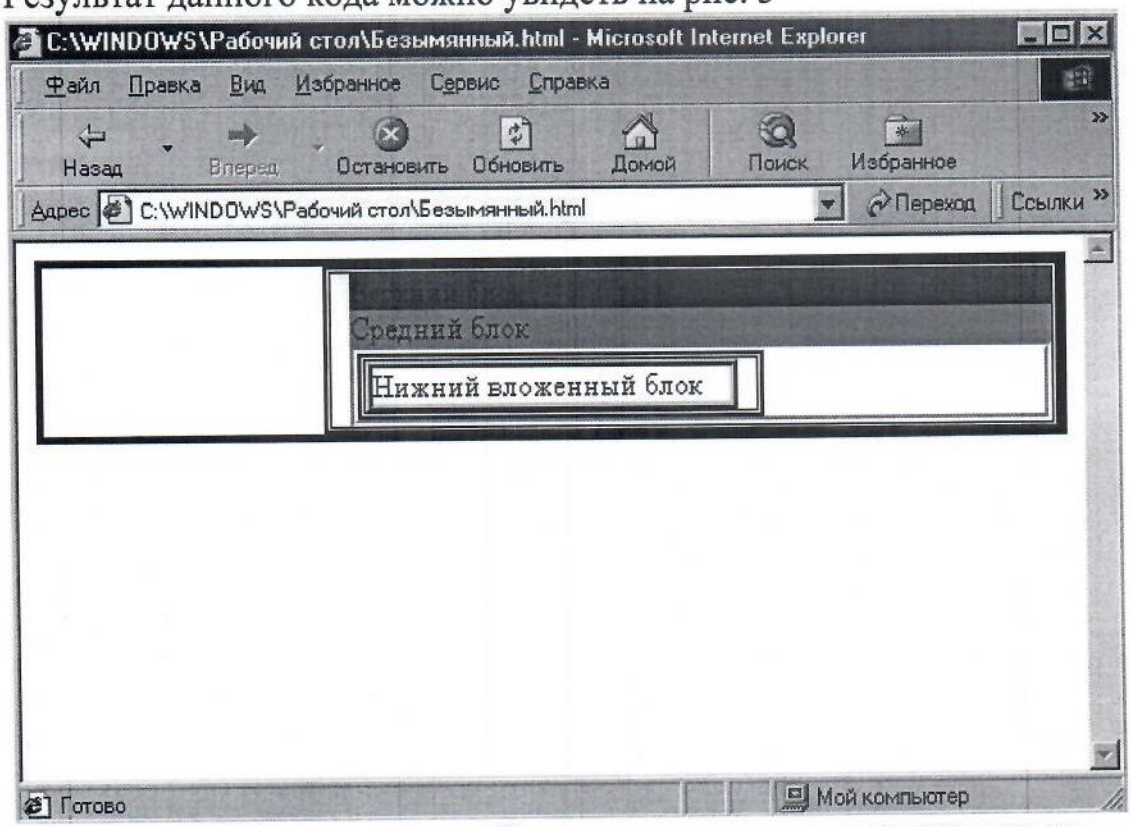

Рис.3 Пример задания блоков с помощью вложенных слоев

Задание

С помощью каскадных таблиц стилей и свойств форматирования блоков задайте страницу в виде таблицы с тремя строками и тремя столбцами. Столбцы и строки должны быть одинаковыми по ширине и высоте соответственно. Задать внешнюю границу и границы между столбцами.

6 РАСПРЕДЕЛЕНИЕ ЧАСОВ ПО ТЕМАМ

Название темы	Содержание задания	Количество часов
Тема 2.12 Что такое CSS? Оформление текста с помощью CSS	Применение CSS для индивидуального сайта	1
Тема 2.13. Цвет и фоновое изображение CSS	Оформление индивидуального сайта	1
Тема 2.14. Модель компоновки	Применение команд CSS для компоновки элементов индивидуального сайта	1
Тема 2.15. Оформление списков и ссылок CSS	Добавление списков и ссылок CSS на индивидуальный сайт	2
Тема 2.16. Оформление таблиц с помощью CSS	Добавление таблиц на индивидуальный сайт	1
Тема 2.17. Позиционирование в CSS	Применение команд CSS для позиционирования элементов индивидуального сайта	1