

Основы веб-технологий. ВолгГТУ. Кафедра ПОАС.

Составитель Овчинников С.А.

## **Лекция 1 Введение в веб-технологии**

## **Лекция 2 Веб-дизайн и юзабилити.**

## **Лекция 3 HTML и CSS.**

### **3.1 Цели и задачи раздела**

Целями изучения данного раздела веб-технологий являются:

- ознакомление технологов с задачами гипертекстовой разметки
- обзор структуры языка, его возможностей, современного состояния, перспектив.
- создание широкой базы знаний и понятий для быстрого освоения более специальных аспектов web-технологий

### **3.2 Определение, задачи**

HTML это стандартный язык разметки документов во Всемирной паутине. Все веб-страницы создаются при помощи языка HTML (или XHTML). Язык HTML интерпретируется браузером и отображается в виде документа, удобном для человека.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без искажений воспроизводиться на оборудовании с различной технической оснащённостью . Однако современное применение HTML очень далеко от его изначальной задачи. С течением времени, основная идея платформонезависимости языка HTML была отдана в своеобразную жертву современным потребностям в мультимедийном и графическом оформлении.

### **3.3 Место языка HTML в иерархии форматов данных**

#### **SGML->HTML**

SGML (англ. Standard Generalized Markup Language ) — метаязык, на котором можно определять язык разметки для документов.

Изначально SGML был разработан для совместного использования машинно-читаемых документов в больших правительственных и аэрокосмических проектах.

HTML является приложением SGML

#### **SGML->XML->XHTML**

XML (англ. eXtensible Markup Language ) — рекомендованный Консорциумом Всемирной паутины язык разметки, фактически представляющий собой свод общих синтаксических правил. XML — текстовый формат, предназначенный для

хранения структурированных данных, для обмена информацией между программами. XML является упрощённым подмножеством языка SGML.

XHTML (англ. Extensible Hypertext Markup Language) — язык разметки веб-страниц, по возможностям сопоставимый с HTML, созданный на базе XML. Соответствует спецификации SGML. Для XHTML можно применять множество технологий, разработанных для XML, например, XSLT и XPath. Анализ XHTML проще и быстрее, чем HTML. Поскольку синтаксис XML строже, чем SGML, обработка XHTML возможна даже на мобильных телефонах с малыми ресурсами

### 3.4 Спецификации

HTML 4.1 - <http://www.w3.org/TR/html401/>

\* Строгий (Strict): не содержит элементов, помеченных как «устаревшие» или «не одобряемые» (deprecated).

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

\* Переходный (Transitional): содержит устаревшие теги в целях совместимости и упрощения перехода со старых версий HTML.

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

### 3.5 Структура, синтаксис, возможности языка HTML

#### 1. Коротко о HTML

HTML (Hyper Text Markup Language) означает язык разметки гипертекста. Этот язык был разработан Тимом Бернерсом-Ли в рамках создания проекта распределенной гипертекстовой системы, которую он назвал World Wide Web (WWW) или Всемирная паутина. HTML предназначен для написания гипертекстовых документов, публикуемых в World Wide Web. Документ на языке HTML может включать следующие компоненты:

- стилизованный и форматированный текст,
- команды включения графических и звуковых файлов,
- гиперсвязи с различными ресурсами Internet.
- скрипты на языке JavaScript и VBScript.
- различные объекты, например Flash-анимацию

Документы HTML являются обычными текстовыми файлами, содержащими специальные теги (или управляющие элементы) разметки. Теги разметки указывают браузеру Web, как надо вывести страницу.

#### Пример

```
<html>
```

```
<head>
<title>Это заголовок страницы</title>
</head>
<body>
<h1>Здравствуйтесь!</h1>
<p>Это моя первая страница HTML. <b>Этот текст выводится жирным
шрифтом.</b></p>
</body>
</html>
```

## Разбор примера

HTML-документ начинается с тега `<html>`, который сообщает браузеру о начале документа HTML и заканчивается тегом `</html>`, который информирует браузер о достижении конца документа HTML.

Текст между тегами `<head>` и `</head>` является информацией заголовка документа. Эта информация не выводится в окне браузера.

Текст "Это заголовок страницы" между тегами `<title>` и `</title>` является заголовком документа. Этот заголовок выводится в строке заголовка окна браузера.

Текст между тегами `<body>` и `</body>` является текстом, который будет выведен в окне браузера. Текст "Здравствуйтесь!" между тегами `<h1>` и `</h1>` будет отображен стилем заголовка, обычно жирным шрифтом большего размера.

Тег `<p>` означает, что начинается новый параграф, тег `</p>` означает конец параграфа.

Текст "Этот текст выводится жирным шрифтом." между тегами `<b>` и `</b>` будет выведен жирным шрифтом.

## 2. Теги

Теги HTML используются для выделения элементов HTML. Обычно теги HTML используются парами и заключены между двумя символами угловых скобок `<` (начальный тег) и `>` (конечный тег). Текст между начальным и конечным тегами является содержимым элемента. Некоторые теги не имеют конечного, например, тег принудительного переноса строки `<br>`, для таких тегов рекомендуется использовать следующее написание `<br />`.

Регистр символов для отображения тегов не важен, например, `<p>` и `<P>` означает одно и то же. Рекомендуется использовать нижний регистр для написания тегов.

### Атрибуты тегов

Теги могут иметь атрибуты, которые предоставляют дополнительную информацию об элементах HTML. Атрибуты всегда используются в виде пары "имя/значение". Общий формат задания атрибутов имеет вид:

```
<имя_тега имя_атрибута="значение">
```

Например, тег:

```
<body bgcolor="red">
```

означает, что цвет фона страницы должен быть красным.

А тег:

```
<p align="center">
```

означает, что параграф необходимо выровнять по центру страницы отображения браузера.

Атрибуты всегда помещаются в начальном теге элемента HTML. Значения атрибутов всегда нужно заключать в кавычки. Чаще используются двойные кавычки, но одиночные кавычки также допустимы.

В некоторых редких ситуациях, когда, например, значение атрибута само содержит кавычки, необходимо использовать одиночные кавычки

Кроме атрибутов, записываемых вышеописанным способом, для некоторых элементов определены специальные флаги, которые просто указываются как <тег имя\_флага>.

```
<input type="text" name="blocked"
value="Пример поля ввода у которого указан флаг readonly"
readonly size="100"><br/><br/>
```

## Основные теги HTML

### Параграфы

Следующий пример показывает, как отображаются параграфы

```
<html>
<body>
<p>Это параграф 1.</p>
<p>Это параграф 2.</p>
<p>Это параграф 3.</p>
</body>
</html>
```

В разных браузерах на разных мониторах с разным разрешением страница будет отображаться по-разному, поэтому не стоит форматировать при помощи добавления пустых строк и пробелов. Любое число пробелов заменяется одним.

Использование пустых параграфов <p> для вставки пустых строк является плохим стилем, вместо этого используйте тег <br/>.

### Заголовки.

Заголовки определяются с помощью тегов от <h1> до <h6>. <h1> определяет заголовок самого большого размера, а <h6> определяет заголовок самого маленького размера.

```
<h1>Это заголовок первого уровня</h1>
<h2>Это заголовок второго уровня</h2>
<h3>Это заголовок третьего уровня</h3>
<h4>Это заголовок четвертого уровня</h4>
<h5>Это заголовок пятого уровня</h5>
<h6>Это заголовок шестого уровня</h6>
```

Заголовки автоматически отделяются дополнительными промежутками от остальных элементов документа.

## Переносы строк.

Для переноса внутри параграфа используется тег `<br>`, который выполняет принудительный перенос строки.

```
<html>
<body>
<p>Это <br>пара<br>граф с переносами строк</p>
</body>
</html>
```

Тег `<br>` не имеет закрывающего тега. Поэтому используется следующее написание тега `<br/>`

## Комментарии в HTML

Тег комментария используется для вставки комментариев в исходный код HTML. Комментарии будут проигнорированы браузером. Комментарии можно использовать для пояснения кода, что может помочь при редактировании исходного кода в будущем.

```
<!-- Это комментарий -->
```

### Таблица основных тегов HTML

Тег	Описание
<code>&lt;html&gt;</code>	Определяет документ HTML
<code>&lt;body&gt;</code>	Определяет основную часть или тело документа
<code>&lt;h1&gt; -- &lt;h6&gt;</code>	Определяет заголовки с 1 по 6
<code>&lt;p&gt;</code>	Определяет параграф
<code>&lt;br&gt;</code>	Вставляет единичный перенос строки
<code>&lt;!--&gt;</code>	Определяет комментарий

## 3. Форматирование текста

```
<html>
<body>
<p>
Если необходимо чтобы к тексту было применено какое-либо форматирование, например, выделение <b>полужирным</b>
```

или `<i>`курсивом`</i>`, необходимо использовать соответствующие теги форматирования. При этом формируемый текст помещается между тегами.

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

**Гипертекстовые ссылки необходимы для соединения с другими документами в Web. Для их записи используется тег `<a>`, который называют "якорь" (anchor).**

```
<a href="page1.htm">
```

Этот текст`</a>` является ссылкой на страницу на этом Web-сайте.

Изображение в качестве ссылки

Этот пример показывает, как использовать в качестве ссылки изображение.

```
<a href="http://www.intuit.ru/">

</a>
```

## Описание тега гипертекстовой ссылки

С помощью атрибута `url` можно задавать ссылки не только по протоколу HTTP, но и по другим:

- `http://...` - создает ссылку на `www`-документ;
- `ftp://...` - создает ссылку на `ftp`-сайт или расположенный на нем файл;
- `mailto:...` - запускает почтовую программу-клиент с заполненным полем имени получателя. Если после адреса поставить знак вопроса, то можно указать дополнительные атрибуты, разделенные знаком `"&"`;

Дополнительные примеры

### *Открытие ссылки в новом окне браузера*

Этот пример показывает, как открыть ссылку на другую страницу в новом окне, чтобы посетителю не нужно было покидать ваш Web-сайт.

```
<a href="lastpage.htm" target="_blank">Последняя страница</a>
```

### *Создание ссылки `mailto`*

Этот пример показывает, как написать письмо на указанный ящик

```
<a
href="mailto:help@intuit.ru&cc=orders@intuit.ru&bcc=admin@intuit.ru?subject=Тестовый%20запрос!">
отправить запрос</a>
```

## Таблицы.

Основным тегом для обозначения таблицы является <table>. Элемент TABLE представляет собой тег-контейнер, в котором размещается содержимое таблицы. Построение таблицы осуществляется по строкам, для обозначения которых применяется контейнер TR. Внутри контейнера строк помещаются контейнеры для обозначения ячеек. Стандарт HTML определяет два типа контейнеров для обозначения ячеек <th> и <td>. Ячейка данных может содержать текст, изображения, списки, параграфы, формы, горизонтальные линейки, таблицы и т. д.

## Примеры

Рассмотрим вышеизложенное на примерах различного вида таблиц.

```
<html>
<body>

<p>
Каждая таблица начинается с тега table.
Каждая строка таблицы начинается с тега tr.
Каждый элемент данных таблицы начинается с тега td.
</p>

<h1>Это пример простейшей таблицы, содержащей одну строку и одну ячейку.</h1>
<table border="1">
<tr>
  <td>Одна строка и одна ячейка</td>
</tr>
</table>

<h1>Одна строка и три столбца:</h1>
<table border="1">
<tr>
  <td> столбец 1</td>
  <td> столбец 2</td>
  <td> столбец 3</td>
</tr>
</table>

<h1>Две строки и три столбца:</h1>
<table border="1">
<tr>
  <td>1.1</td>
  <td>1.2</td>
  <td>1.3</td>
</tr>
<tr>
  <td>2.1</td>
  <td>2.2</td>
  <td>2.3</td>
</tr>
</table>

</body>
</html>
```

```
<h1>Рамка таблицы</h1>
```

```
<html>  
<body>  
<h1>Обычная рамка:</h1>  
<table border="1">  
<tr>  
  <td>Первая</td>  
  <td>строка</td>  
</tr>  
<tr>  
  <td>Вторая </td>  
  <td>строка</td>  
</tr>  
</table>
```

```
<h1>Толстая рамка:</h1>  
<table border="10">  
<tr>  
  <td>Первая</td>  
  <td>строка</td>  
</tr>  
<tr>  
  <td>Вторая </td>  
  <td>строка</td>  
</tr>  
</table>
```

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

```
<table border="1">  
<tr>  
<td>строка 1, ячейка 1</td>  
<td>строка 1, ячейка 2</td>  
</tr>  
<tr>  
<td>строка 2, ячейка 1</td>  
<td>строка 2, ячейка 2</td>  
</tr>  
</table>
```

Для тега table определены следующие атрибуты.

- align - определяет способ горизонтального выравнивания таблицы на странице. Возможные значения: left, center, right. Значение по умолчанию - left.
- valign - должен определять способ вертикального выравнивания для содержимого таблицы. Возможные значения: top, bottom, middle.
- border - определяет ширину внешней рамки таблицы (в пикселах). При BORDER="0" или при отсутствии этого параметра рамка отображаться не будет.
- cellpadding - определяет расстояние (в пикселах) между рамкой каждой ячейки таблицы и содержащимся в ней материалом.
- cellspacing - определяет расстояние (в пикселах) между границами соседних ячеек.
- width - определяет ширину таблицы. Ширина задается либо в пикселах, либо в процентном отношении к ширине окна браузера. По умолчанию этот

параметр определяется автоматически в зависимости от объема содержащегося в таблице материала.

- `height` - определяет высоту таблицы. Высота задается либо в пикселах, либо в процентном отношении к высоте окна браузера. По умолчанию этот параметр определяется автоматически в зависимости от объема содержащегося в таблице материала.
- `bgcolor` - определяет цвет фона ячеек таблицы. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.
- `background` - позволяет заполнить фон таблицы рисунком. В качестве значения необходимо указать URL рисунка.

Ячейки таблицы без содержимого в большинстве браузеров выводятся не очень хорошо. Если необходимо сделать ячейку таблицы пустой, то разместите в ней неразрывный пробел `&nbsp;`;

Для тегов `<td>` и `<th>` очень полезными являются атрибуты `colspan` и `rowspan`. Первый показывает сколько ячеек надо объединить по горизонтали, а второй по вертикали. Следует заметить, что если мы применяем объединение ячеек, то общее число ячеек с учетом объединенных должно быть равным. Продемонстрируем на примере.

```
<html>
<body>

<h4>Правильное применение colspan:</h4>
<table border="1">
<tr>
  <td colspan="2">100</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>
  <td>500</td>
  <td>600</td>
</tr>
</table>

<h4>Неправильное применение colspan:</h4>
<table border="1">
<tr>
  <td colspan="2">100</td>
  <td>200</td>
  <td>300</td>
</tr>
<tr>
  <td>400</td>
  <td>500</td>
  <td>600</td>
</tr>
</table>

<h4>Правильное применение rowspan:</h4>
<table border="1">
<tr>
  <td rowspan="2">100</td>
```

```

        <td>200</td>
        <td>300</td>
</tr>
<tr>
    <td>500</td>
    <td>600</td>
</tr>
</table>

<h4>Неправильное применение rowspan:</h4>
<table border="1">
<tr>
    <td rowspan="2">100</td>
    <td>200</td>
    <td>300</td>
</tr>
<tr>
    <td>400</td>
    <td>500</td>
    <td>600</td>
</tr>
</table>
</body>
</html>

```

HTML поддерживает упорядоченные списки, неупорядоченные списки, и списки определений. Отличаются эти разновидности списков лишь способом оформления. Перед пунктами неупорядоченных списков обычно ставятся символы-буллеты (bullets), например, точки, ромбики и т.п., в то время как пунктам упорядоченных списков предшествуют их номера.

## Неупорядоченные списки

Неупорядоченный список является списком элементов. Элементы списка маркируются с помощью специальных знаков (обычно небольшой черный круг).

Неупорядоченный список начинается с тега <ul>. Каждый элемент списка начинается с тега <li>.

```

<ul>
    <li>элемент 1</li>
    <li>элемент 2</li>
    <li>элемент 3</li>
</ul>

```

Внутри элемента списка можно помещать параграфы, переносы строк, изображения, ссылки, другие списки, и т.д.

## Упорядоченные списки

Упорядоченный список также является списком элементов. Элементы списка маркируются с помощью чисел или букв.

Упорядоченный список начинается с тега `<ol>`. Каждый элемент списка начинается с тега `<li>`. У тега `<ol>` может быть два атрибута `start` (определяет первое число, с которого начинается нумерация пунктов) и `type` (определяет стиль нумерации пунктов). Может иметь значения:

- "A" - заглавные буквы A, B, C ...
- "a" - строчные буквы a, b, c ...
- "I" - большие римские числа I, II, III ...
- "i" - маленькие римские числа i, ii, iii ...
- "1" - арабские числа 1, 2, 3 ...

```
<ol>
  <li>элемент 1</li>
  <li>элемент 2</li>
  <li>элемент 3</li>
</ol>
```

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

Внутри элемента списка можно помещать параграфы, переносы строк, изображения, ссылки, другие списки, и т.д.

## Теги списков

Тег	Описание
<code>&lt;ol&gt;</code>	Определяет упорядоченный список
<code>&lt;ul&gt;</code>	Определяет неупорядоченный список
<code>&lt;li&gt;</code>	Определяет элемент списка
<code>&lt;dl&gt;</code>	Определяет список определений
<code>&lt;dt&gt;</code>	Определяет термин определения
<code>&lt;dd&gt;</code>	Определяет описание определения
<code>&lt;dir&gt;</code>	Не рекомендуется. Используйте вместо этого <code>&lt;ul&gt;</code>
<code>&lt;menu&gt;</code>	Не рекомендуется. Используйте вместо этого <code>&lt;ul&gt;</code>

## 4. Формы

**Формы HTML предназначены для организации взаимодействия с пользователем.**

Формы позволяют вводить текст, осуществлять выбор из предложенных значений при помощи списков или кнопок. С помощью форм можно организовать интерактивный обмен информацией между Web-страницей и сервером. Можно определить формы как электронные бланки для заполнения различных данных таких как, например, имя, возраст, выбор страны проживания и других. Как

правило, форма работает совместно с установленным на сервере сценарным приложением, обрабатывающим введенную информацию.

## Примеры форм

### *Форма поиска*

Одна из наиболее распространенных форм. Пользователь получает возможность ввести искомый запрос, определить область поиска и отправить запрос на сервер для обработки.

```
<form name="input" action="html_form_action.asp" method="get">
  <table border=1 bgcolor="#ddffdd">
    <tr>
      <td align="center">
        <input type="text" name="search" size="50" value="Строка для поиска">
        <input type="submit" value="Поиск">
        <br>
        <input type="checkbox" name="news">Искать в новостях
        <input type="checkbox" name="arhive">Искать в архивах
      </td>
    </tr>
  </table>
</form>
```

### *Формы*

Форма является областью, которая может содержать элементы, позволяющие пользователю вводить информацию (такие как текстовые поля, поля многострочного текста, раскрывающиеся меню, переключатели, флажки, и т.д.).

Форма определяется с помощью тегов `<form></form>`, между которыми располагаются поля ввода, кнопки, а также все необходимые элементы оформления формы.

Тег `<form>` имеет ряд атрибутов, из которых необходимо выделить атрибуты `action` и `method`. Без этих атрибутов форма не сможет передать информацию от пользователя на сервер.

```
<form action="html_form_action.asp" method=get>
```

Атрибут `Action` указывает URL-адрес объекта, который должен получить данные формы.

Атрибут `method` может иметь два значения: `get` и `post`.

Значение атрибута `method=get` заставляет Web-браузер передать все данные формуляра по URL-адресу, заданному в `action`. При этом введенные при

заполнении формы данные просто добавляются в адресную строку с использованием разделителя – знака вопроса. Этот метод удобен для небольших форм.

Значение атрибута `method=post` заставляет Web-браузер, прежде всего, связаться с сервером, обрабатывающим форму, и только после установки связи приступить к передаче данных, для обработки которых будут использоваться специальные сценарии.

## Поля ввода

Большинство элементов ввода и управления в форме можно описать при помощи тега `<input>`, обязательными для которого являются атрибуты `name` (приписывает данному элементу ввода уникальное имя, используемое для дальнейшей обработки формы) и `type` (определяет тип элемента управления или ввода).

### *Текстовые поля*

Текстовое поле (`type=text`) определяет однострочное поле ввода и используется, когда необходимо, чтобы пользователь ввел в форму данные в произвольной форме но ограниченные по объему (слова, словосочетания, числа и т.д.). Следующий пример демонстрирует простейшую форму для ввода имени и фамилии.

```
<html>
<body>

<form>
  Имя:
  <input type="text" name="firstname">
  <br>
  Фамилия:
  <input type="text" name="lastname">
</form>

</body>
</html>
```

Здесь необходимо отметить, что атрибут `size` ограничивает только видимую область ввода данных, а не длину вводимой строки. Для этой цели используется атрибут `maxlength`.

Необходимо отметить, что применение атрибутов для различных типов полей вывода и управления может отличаться. Так, например, атрибут `size` для текстовых полей ввода (`type="text"` или `type="password"`) указывает максимальное количество символов, отображаемых в поле, а для прочих элементов – занимаемый по горизонтали размер в пикселях.

## ***Поле пароля***

Поле пароля (`type=password`) создает защищенное поле ввода, которое дает возможность пользователю, заполняющему форму ввести текст, но в отличие от обычного текстового поля, вводимые данные при отображении на мониторе заменяются звездочками или точками.

Следующий пример демонстрирует простейшую форму для ввода имени пользователя и пароля.

```
<form>
  Имя пользователя:
  <input type="text" name="user" value="Елена">
  <br>
  Пароль:
  <input type="password" name="password" value="tktyf">
</form>

</body>
</html>
```

Необходимо обратить внимание, что хотя значение пароля и задано по умолчанию, при отображении браузер выводит вместо символов звездочки.

## ***Переключатели***

Переключатели или радиокнопки (`type=radio`) определяют поля выбора одного значения из нескольких доступных. Поля этого типа часто используются в диалоговых окнах. Для каждой позиции переключателя создается свой тег `<input type=radio>`. Группируются переключатели при помощи одинакового имени, задаваемого атрибутом `name`.

```
<html>
<body>

<form>
  Укажите Ваш пол:
  <br>
  <input type="radio" name="sex" value="male"> мужчина
  <br>
  <input type="radio" name="sex" value="female"> женщина
</form>

</body>
</html>
```

Необходимо отметить, что в отличие от текстового поля и поля пароля атрибут `value` задает значение, которое будет передано серверу для дальнейшей обработки в случае выбора данного переключателя. Для выбора по умолчанию одного из возможных значений группы переключателей используется атрибут `checked`.

```
<html>
```

```
<body>

<form>
  Укажите Ваш пол:
  <br>
  <input type="radio" name="sex" value="male" checked="checked"> мужчина
  <br>
  <input type="radio" name="sex" value="female"> женщина
</form>

</body>
</html>
```

В данном примере по умолчанию выбран мужской пол:

### ***Флажки***

Флажки (`type=checkbox`) используются, когда необходимо, чтобы пользователь выбрал один или несколько вариантов из ограниченного числа вариантов выбора. Флажки в форме не зависят друг от друга, их можно установить или сбросить в любой комбинации. Для каждого флажка необходимо задать свое уникальное имя при помощи атрибута `name`. Создание двух флажков с одним именем не вызовет ошибки при отображении формы, но не позволит сценарию обработки на сервере корректно обработать передаваемые с формы данные.

```
<html>
<body>

<form>
  В этом году я собираюсь приобрести:
  <br>
  <input type="checkbox" name="computer">
  Компьютер
  <br>
  <input type="checkbox" name="notebook">
  Ноутбук
  <br>
  <input type="checkbox" name="printer">
  Принтер
  <br>
  <input type="checkbox" name="scanner">
  Сканер
</form>

</body>
</html>
```

При помощи атрибута `checked` можно установить, какие из флажков будут выбраны по умолчанию при загрузке страницы. Отличие от переключателей заключается только в том, что для флажков можно отметить сразу несколько вариантов.

```
<html>
```

```
<body>

<form>
  В этом году я собираюсь приобрести:
  <br>
  <input type="checkbox" name="computer" checked="checked">
  Компьютер
  <br>
  <input type="checkbox" name="notebook">
  Ноутбук
  <br>
  <input type="checkbox" name="printer">
  Принтер
  <br>
  <input type="checkbox" name="scanner" checked="checked">
  Сканер
</form>

</body>
</html>
```

В данном примере по умолчанию выбраны флажки "Компьютер" и "Сканер":

При отправке данных формы с флажками на сервер выбранным флажкам присваивается значение по умолчанию "on". Как правило, этого достаточно для корректной обработки данных, но в некоторых случаях удобнее задать каждому флажку свое значение при помощи атрибута value.

```
<input type="checkbox" name="printer" value="Принтер">
```

### ***Командные кнопки***

Командная кнопка отправки (type=submit) используется для выполнения пересылки данных формы на сервер. Метод отправки и адрес файла, обрабатывающего полученную информацию задаются в теге <form> при помощи атрибутов method и action. Командная кнопка сброса (type=reset) возвращает форму к исходному состоянию (очищает форму). При этом данные не передаются.

В следующем примере показана форма поиска с двумя кнопками отправки и сброса.

```
<html>
<body>

  <form name="input" action="html_form_action.asp" method="get">
  Найти:
  <input type="text" name="search" size=25>
  <br>
  <input type="submit">
  <input type="reset">

</form>

</body>
```

```
</html>
```

Если ввести в текстовое поле какие-то символы и нажать кнопку "Подача запроса", то введенная информация будет послана на страницу с именем "html\_form\_action.asp". При нажатии на кнопку "Сброс" текстовое поле очистится.

Надписи на кнопках "Подача запроса" и "Сброс" установлены по умолчанию. Для их изменения необходимо использовать атрибут value.

Кроме кнопок отправки и сброса существует также возможность добавлять пользовательские кнопки (type=button), которые могут использоваться для выполнения процедур (скриптов) непосредственно на Web-странице.

### ***Поле выбора файла***

Поле выбора файла (type=file) создает поле для выбора файла, который будет загружен на сервер вместе с информацией формы. Рядом с полем ввода отображается командная кнопка "Обзор...", открывающая стандартное диалоговое окно выбора файла.

```
<html>
<body>

<form>
  Прикрепить файл:
  <br>
  <input type="file" size="50">
</form>

</body>
</html>
```

Для поля выбора файла по аналогии с текстовым полем можно использовать атрибуты size, maxlength, value.

### ***Списки выбора***

Списки выбора бывают двух типов: раскрывающиеся списки (выпадающие меню) и списки с множественным выбором. Независимо от типов списков описываются они одинаково с помощью пары тегов <select> </select>. Отдельные элементы списка задаются с использованием тега <option>. Тип списка определяется при помощи атрибутов тега <select>.

Атрибут name задает имя поля для отправки выбранных пунктов списка на сервер. Атрибут multiple разрешает множественный выбор. Атрибут size определяет, какое количество пунктов списка будет одновременно отображено на экране. При этом,

если атрибут `multiple` не задан и `size=1`, то на экране отображается раскрывающийся список, если же задан атрибут `Multiple` или значение `size` больше 1, то список отображается развернутым.

### Раскрывающийся список выбора

Раскрывающиеся списки выбора по своему назначению соответствуют переключателям, в то же время их использование предпочтительно в тех случаях, когда количество вариантов выбора достаточно велико. Как правило, при выборе более чем из трех вариантов желательно вместо переключателей использовать раскрывающиеся списки.

В следующем примере создан простой раскрывающийся список выбора ноутбука по производителю.

```
<html>
<body>

<form>
  Выбор ноутбука по производителю:
  <select name="notebook">
    <option value="acer">Acer
    <option value="asus">Asus
    <option value="compaq">Compaq
    <option value="hp">HP
    <option value="sony">Sony
    <option value="toshiba">Toshiba
  </select>
</form>

</body>
</html>
```

Необходимо заметить, что по умолчанию выбирается первое значение из списка. При помощи атрибута `selected` тега `<option>` это значение можно изменить. Следующий пример показывает раскрывающийся список выбора размера экрана ноутбука с предварительно установленным значением "15.4".

```
<html>
<body>

<form>
  Выбор размера экрана ноутбука
  <select name="tft">
    <option value="tft-12">12"
    <option value="tft-13">13"
    <option value="tft-14">14"
    <option value="tft-15">15"
    <option value="tft-15-4" selected="selected">15.4"
    <option value="tft-17">17"
  </select>
</form>
```

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

### **Текстовая область**

В отличие от текстового поля `<input type=text>` текстовая область позволяет вводить многострочный текст большого объема. Такие области часто используются при вводе сообщений, комментариев.

Текстовая область описывается при помощи тегов `<textarea>` `</textarea>`, между которыми можно разместить предварительно отформатированный стандартный текст. Атрибуты `cols` и `rows` задают размер видимой области текстового поля.

В следующем примере создана текстовая область с предварительно введенным текстом.

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

```
  <textarea rows="7" cols="30">
```

В данном примере мы создали текстовую область с шириной в 30 символов и высотой в 7 строк.

Заданное значение высоты не ограничивает общий объем вводимого текста, а влияет только на размер отображаемой на экране текстовой области.

Для просмотра всего текста необходимо воспользоваться полосой прокрутки.

```
  </textarea>
```

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

### Теги форм

Тег	Описание
<code>&lt;form&gt;</code>	Определяет форму для ввода пользователя
<code>&lt;input&gt;</code>	Определяет поле ввода
<code>&lt;textarea&gt;</code>	Определяет текстовую область (элемент управления для ввода многострочного текста)
<code>&lt;label&gt;</code>	Определяет метку для элемента управления
<code>&lt;fieldset&gt;</code>	Определяет набор полей
<code>&lt;legend&gt;</code>	Определяет заглавие для набора полей
<code>&lt;select&gt;</code>	Определяет список выбора (раскрывающееся поле)
<code>&lt;optgroup&gt;</code>	Определяет группу вариантов выбора
<code>&lt;option&gt;</code>	Определяет вариант в раскрывающемся поле
<code>&lt;button&gt;</code>	Определяет кнопку
<code>&lt;isindex&gt;</code>	Не рекомендуется. Используйте вместо этого <code>&lt;input&gt;</code>

## 5. Мультимедиа

**В HTML предусмотрен целый ряд возможностей для работы с мультимедиа. Это встраивание графики и использование звуков, применение анимационных роликов и видеофильмов.**

Применение мультимедийных возможностей является далеко не последним аргументом в борьбе за привлечение новых пользователей web-сайта, в то же время при использовании мультимедиа необходимо соблюдать чувство меры. Хорошо оформленная web-страница позволит привлечь и удержать внимание посетителей, в то же время чрезмерное увлечение изображениями, звуковыми эффектами и т.п. может затруднить просмотр страницы (при медленных каналах связи) или отпугнуть чересчур навязчивыми мелодиями.

В общем случае при использовании мультимедиа на web-страницах желательно придерживаться следующих правил: оформление сайта должно соответствовать его содержанию, применение графики и мультимедиа должно упрощать навигацию по сайту, должна быть предусмотрена возможность работы с web-страницей для медленных каналов связи (замещающие надписи, дублирующие текстовые меню или специальные текстовые варианты страницы).

В рамках данного курса будут рассмотрены возможности HTML по использованию изображений на web-страницах.

## Примеры

### *Вставка изображений*

```
<html>
<body>

<p>
Изображение:

</p>

<p>
Динамическое изображение:

</p>

<p>
Обратите внимание, что синтаксис вставки динамического изображения
не отличается от синтаксиса для обычного изображения.
</p>

</body>
</html>
```

Этот пример показывает, как вывести изображения на Web-странице.

## ***Вставка изображений из различных мест***

```
<html>
<body>

<p>
Изображение из другой папки:

</p>

<p>
Изображение с сайта ИНТУИТ:

</p>

</body>
</html>
```

Для выравнивания изображений используется атрибут align.

- align=top – изображение выравнивается по верхнему краю текущей текстовой строки, не меняя позиции по горизонтали. При этом речь идет как о тексте, так и о графике;
- align=middle – изображение центрируется по вертикали на базовой линии текущей текстовой строки, не меняя позицию по горизонтали;
- align=bottom – нижний край изображения выравнивается по горизонтали на базовой линии текущей текстовой строки;
- align=left – изображение смещается к левому краю рабочей зоны, последующий текст сразу же начинает "обтекать" графику;
- align=right – то же что и для left, только изображение смещается к правой части рабочей зоны.

### **Атрибут Alt**

В ряде случаев графическое изображение на Web-странице не может быть отражено. Во избежание этого в HTML используется атрибут alt, который задает цепочку символов (максимальная длина 1024 символа), отображаемую в браузере вместо изображения и в произвольной форме описывающую его (заменяющий текст).

Использование атрибута alt считается признаком хорошего HTML-тона.

## **3.6 Карты изображений**

Наряду с использованием изображений в качестве иллюстраций к тексту или элементов оформления Web-страницы, в HTML предусмотрена возможность создания карт изображений (imagemap), отдельные области которого могут являться гиперссылками на другие разделы или страницы Web-сайта. В общем виде это соответствует использованию изображения в качестве гиперссылки с тем отличием, что на одной карте изображений можно создать несколько несовпадающих областей, и соответственно, гиперссылок.

Применение таких карт изображений удобно для создания географических справочных систем, путеводителей, карт погоды. Также карты изображений находят широкое применение при создании сложных графических меню.

В HTML предусмотрены два варианта обработки карт изображений: обработка карты изображения непосредственно браузером и передача на сервер координат указателя мыши для дальнейшей обработки.

### ***Карты изображений, обрабатываемые браузером***

Автономные (обрабатываемые браузером) карты изображений описываются с помощью атрибута usemap тега <img>:

```

```

Где "figure\_1.png" - имя файла, содержащего изображение, "#coordinates" - ссылка на часть HTML-документа, описывающего координатные области.

Координатные области карты изображений определяются при помощи тегов <map> и <area>.

Тег <area> описывает координаты отдельной области изображения, параметры которой и адрес гиперссылки задаются при помощи атрибутов shape, coords и href.

Атрибут shape определяет форму области-ссылки. По умолчанию ему присваивается значение shape="rect" - прямоугольник. Также область может быть описана в форме окружности (shape="circle") или многоугольника (shape="poly").

Атрибут coords определяет размеры области. В зависимости от типа размечаемой области может меняться значение этого атрибута. Так для shape="rect" указываются две пары координат (левого верхнего и правого нижнего угла прямоугольника) в пикселях. Для shape="circle" указываются координаты центра окружности и ее радиус, а для shape="poly" последовательно указываются координаты вершин многоугольника.

Атрибут href задает URL-адрес ссылки для перехода в случае щелчка мышью на выбранной области.

Заданные при помощи тегов <area> координатные области изображения ограничиваются при помощи тегов <map>-</map>.

#### Теги изображений

Тег	Описание
<img>	Определяет изображение
<map>	Определяет карту ссылок

`<area>` Определяет активную область внутри карты ссылок

### 3.7 Цвета в HTML

Цвета выводятся с помощью смешения источников RED (красного), GREEN (зеленого), и BLUE (синего) цвета.

#### Значения цветов

Цвета определяют с помощью шестнадцатеричной записи комбинации значений красного, зеленого и синего цветов (RGB). Наименьшее значение, которое можно задать одному из источников равно 0 (hex #00). Максимальное значение равно 255 (hex #FF).

#### Имена цветов

Некоторая совокупность названий цветов поддерживается большинством браузеров.

Примечание: Только 16 названий цветов поддерживается стандартом W3C для HTML 4.0 (aqua (голубой), black (черный), blue (синий), fuchsia (фуксия), gray (серый), green (зеленый), lime (лайм), maroon (темно-бордовый), navy (темно-синий), olive (оливковый), purple (сиреневый), red (красный), silver (светло-серый), teal (сине-зеленый), white (белый), и yellow (желтый)).

#### Компоновка документа в HTML

Выше были рассмотрены основные теги HTML. Используя их, уже можно создавать свои страницы. Но создание страницы это не только верстка материалов, но и компоновка всех элементов (меню, заголовки страницы, основное информационное наполнение, баннеры и др.) на странице HTML документа.

Одним из наиболее распространенных способом компоновки страницы является использование таблиц. Этот способ многими специалистами считается устаревшим и, более того, идеологически неверным.

Более современный способ – верстка с помощью слоев.

При нем таблицы используются, но исключительно для табличных данных. При верстке слоями широко используются возможности CSS.

<http://www.htmlbook.ru/content/?pid=17>

<http://www.htmlbook.ru/content/?id=83>

#### Использование шрифтов в HTML

Тег `<font>` в HTML использовать не рекомендуется.

## Атрибуты тега font

Атрибут	Пример	Назначение
size="число"	size="2"	Определяет размер шрифта
size="+число"	size="+1"	Увеличивает размер шрифта
size="-число"	size="-1"	Уменьшает размер шрифта
face="название шрифта"	face="Times"	Определяет название шрифта
color="значение цвета"	color="#eeff00"	Определяет цвет шрифта
color="название цвета"	color="red"	Определяет цвет шрифта

Тег <font> НЕ должен использоваться.

Тег <font> не рекомендуется использовать в последних версиях HTML

Консорциум World Wide Web (W3C) удалил тег <font> из своих рекомендаций.

Правильно будет использовать стили - Примеры

### Задание шрифта текста

```
<html>
<body>
<h1 style="font-family:verdana">Заголовок </h1>
<p style="font-family:courier">Параграф</p>
</body>
</html>
```

### Задание размера шрифта текста

```
<html>
<body>
<h1 style="font-size:150%">Заголовок</h1>
<p style="font-size:80%">Параграф</p>
</body>
</html>
```

### Задание цвета шрифта текста

```
<html>
<body>
<h1 style="color:blue">Заголовок </h1>
<p style="color:red">Параграф</p>
</body>
</html>
```

## Задание для текста шрифта, его размера и цвета

```
<html>
<body>
<p style="font-family:verdana;font-size:80%;color:green">
Это параграф, содержащий некоторый текст. Это текст, содержащийся в
параграфе.
Это все тот-же параграф с текстом.
</p>
</body>
</html>
```

## Валидация HTML-документа

Документ HTML проверяется согласно Определению типа документа (DTD).  
Прежде чем файл HTML можно будет проверить, необходимо добавить в качестве первой строки файла правильный DTD.

DTD Strict (строгий) для HTML 4.01 включает элементы и атрибуты, которые рекомендованы к использованию и не появляются в наборах фреймов:

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

DTD Transitional (переходный) для HTML 4.01 включает все из строгого DTD плюс не рекомендованные элементы и атрибуты:

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

DTD Frameset (набор фреймов) для HTML 4.01 включает все из переходного DTD плюс также фреймы:

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

Проверьте свой файл HTML с помощью средства проверки W3C

Введите адрес своей страницы в поле формы, показанной ниже:

```
<html>
<head>
<form method="get" action="http://validator.w3.org/check" target="_blank">
Введите адрес своей страницы в расположенное ниже поле <br />
<br />
<input name="uri" size="50" />
<br /><br />
<input type="submit" value="Validate the page" />
</form>
</body>
</html>
```

## 6. Стили

### Использование стилей в HTML

В HTML 4.0 все форматирование можно переместить из документа HTML в отдельную таблицу стилей.

```
<html>
<head>
<style type="text/css">
h1 {color: red}
h3 {color: blue}
</style>
</head>

<body>
<h1>Это заголовок 1</h1>
<h3>Это заголовок 3</h3>
</body>
</html>
```

Этот пример показывает, как форматировать документ HTML с помощью информации о стилях, добавленной в раздел заголовка <head>.

```
<html>
<body>

<a href="lastpage.htm"
style="text-decoration:none">
ЭТО НЕПОДЧЕРКНУТАЯ ССЫЛКА!
</a>

</body>
</html>
```

Этот пример показывает, как с помощью атрибута style сделать ссылку, которая не подчеркивается.

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="styles.css" >
</head>

<body>
<h1>Я отформатирован с помощью присоединенной таблицы стилей</h1>
<p>Я тоже!</p>
</body>
</html>
```

Этот пример показывает, как использовать тег <link> для соединения с внешней таблицей стилей.

### Как использовать стили

Когда браузер считывает таблицу стилей, он форматирует документ согласно этой таблице. Существует три способа использования таблицы стилей.

### ***Внешняя таблица стилей***

Внешняя таблица стилей является идеальной, когда стиль применяется к нескольким страницам. С помощью внешней таблицы стилей можно изменить внешний вид всего Web-сайта, изменяя один файл. Каждая страница должна соединяться с таблицей стилей с помощью тега <link>. Тег <link> находится в разделе заголовка <head>.

```
<html>
<head>
<link rel="stylesheet" type="text/css"
href="mystyle.css">
</head>
</html>
```

### ***Внутренняя таблица стилей***

Внутренняя таблица стилей должна использоваться, когда один документ использует единый стиль. Внутренние стили определяют в разделе заголовка с помощью тега <style>.

```
<html>
<head>
<style type="text/css">
body {background-color: red}
p {margin-left: 20px}
</style>
</head>
</html>
```

### ***Встроенные стили***

Встроенный стиль должен использоваться, когда особый стиль должен быть применен к единственному появлению элемента.

Чтобы использовать встроенные стили, используется атрибут style в соответствующем теге. Атрибут style может содержать любое свойство CSS. Следующий пример показывает, как изменить цвет и левое поле параграфа:

```
<html>
<body>
<p style="color: red; margin-left: 20px">
Это параграф
</p>
</body>
</html>
```

#### Теги заголовка

Тег	Описание
<head>	Определяет информацию о документе
<title>	Определяет заглавие документа

<code>&lt;base&gt;</code>	Определяет базовый URL для всех ссылок на странице
<code>&lt;link&gt;</code>	Определяет ссылку на ресурс
<code>&lt;meta&gt;</code>	Определяет мета-информацию
<code>&lt;!DOCTYPE&gt;</code>	Определяет тип документа. Этот тег идет перед начальным тегом <code>&lt;html&gt;</code> .

## 7. Сценарии в HTML

Сценарии на страницах HTML позволяют сделать их более динамичными и интерактивными.

```
<html>
<body>

<script type="text/javascript">
document.write("<h1>Всем привет!</h1>")
</script>

</body>
</html>
```

### Стандартные атрибуты HTML 4.0

Теги HTML могут иметь атрибуты. Специальные атрибуты для каждого тега перечислены в описании каждого тега, Перечисленные здесь атрибуты являются базовыми и атрибутами языка, которые стандартны для всех тегов (с небольшими исключениями):

### Базовые атрибуты

Не действительны в элементах `base`, `head`, `html`, `meta`, `param`, `script`, `style`, и `title`.

Атрибут	Значение	Описание
<code>class</code>	<code>class_rule</code> или <code>style_rule</code>	Класс элемента
<code>id</code>	<code>номер_id</code>	Уникальный <code>id</code> элемента
<code>style</code>	<code>определение_стиля</code>	Определение встроенного стиля
<code>title</code>	<code>текст_подсказки</code>	Текст, выводимый в качестве подсказки

### Атрибуты языка

Не действительны в элементах `base`, `br`, `frame`, `frameset`, `hr`, `iframe`, `param` и `script`.

Атрибут	Значение	Описание
<code>dir</code>	<code>ltr</code>   <code>rtl</code>	Задаёт направление вывода текста
<code>lang</code>	<code>код_языка</code>	Задаёт код языка

## Атрибуты клавиатуры

Атрибут	Значение	Описание
accesskey	символ	Задает клавишу быстрого доступа для элемента
tabindex	число	Задает для элемента порядок перехода по клавише табуляции

## Атрибуты событий в HTML 4.0

Новым в HTML 4.0 является возможность для событий HTML запускать действия браузера, такие как запуск JavaScript, когда пользователь щелкает на элементе HTML.

Ниже представлен список атрибутов, которые можно использовать в тегах HTML для определения действий событий.

Чтобы больше узнать о программировании с помощью этих событий, почитайте учебники по JavaScript и DHTML.

## Атрибуты событий окна

Действительны только в элементах body и frameset.

Атрибут	Значение	Описание
onload	сценарий	Сценарий, который выполняется при загрузке документа
onunload	сценарий	Сценарий, который выполняется при выгрузке документа

## Атрибуты событий формы

Действительны только в элементах form.

Атрибут	Значение	Описание
onchange	сценарий	Сценарий, который выполняется при изменении элемента
onsubmit	сценарий	Сценарий, который выполняется при отправке формы
onreset	сценарий	Сценарий, который выполняется при сбросе формы
onselect	сценарий	Сценарий, который выполняется при выборе элемента
onblur	сценарий	Сценарий, который выполняется, когда элемент теряет фокус
onfocus	сценарий	Сценарий, который выполняется, когда элемент получает фокус

## Атрибуты событий клавиатуры

Не действительны в элементах base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, title.

<b>Атрибут</b>	<b>Значение</b>	<b>Описание</b>
onkeydown	сценарий	Что делать, когда нажата клавиша
onkeypress	сценарий	Что делать, когда клавиша нажата и отпущена
onkeyup	сценарий	Что делать, когда клавиша отпущена

## Атрибуты событий мыши

Не действительны в элементах base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, title.

Атрибут	Значение	Описание
onclick	сценарий	Что делать при щелчке мыши
ondblclick	сценарий	Что делать при двойном щелчке мыши
onmousedown	сценарий	Что делать, когда нажата клавиша мыши
onmousemove	сценарий	Что делать, когда перемещается указатель мыши
onmouseout	сценарий	Что делать, когда указатель мыши покидает элемент
onmouseover	сценарий	Что делать, когда указатель мыши проходит над элементом
onmouseup	сценарий	Что делать, когда кнопка мыши отпущена

## Справочник по основным элементам HTML

### Структура документа HTML

```
<html>
<head>
<title>Здесь находится название документа</title>
</head>
<body>
Здесь находится выводимый текст
</body>

</html>
```

### Элементы заголовка

```
<h1>Самый крупный заголовок</h1>
<h2> . . . </h2>
<h3> . . . </h3>
<h4> . . . </h4>
<h5> . . . </h5>
<h6>Самый мелкий заголовок</h6>
```

### Элементы текста

```
<p>Это параграф</p>
<br> (перенос строки)
<hr> (горизонтальная линейка)
<pre>Этот текст заранее отформатирован</pre>
```

### Логические стили

```
<em>Этот текст выделен </em>
<strong>Этот текст акцентирован</strong>
<code>Это некоторый программный код</code>
```

### Физические стили

```
<b>Этот текст жирный</b>
<i>Этот текст наклонный</i>
```

## Ссылки, анкеры, и элементы изображений

```
<a href="http://www.example.com/">Это ссылка</a>
<a href="http://www.example.com/"></a>
<a href="mailto:webmaster@example.com">Пошлите e-mail</a>
```

Именованный анкер:

```
<a name="tips">Раздел полезных советов</a>
<a href="#tips">Переход в Раздел полезных советов</a>
```

## Неупорядоченный список

```
<ul>
<li>Первый элемент</li>
<li>Следующий элемент</li>
</ul>
```

## Упорядоченный список

```
<ol>
<li>Первый элемент</li>
<li>Следующий элемент</li>
</ol>
```

## Список определений

```
<dl>
<dt>Первый термин</dt>
<dd>Определение</dd>
<dt>Следующий термин</dt>
<dd>Определение</dd>
</dl>
```

## Таблицы

```
<table border="1">
<tr>
<th>какой-то заголовок</th>
<th>какой-то заголовок</th>
</tr>
<tr>
<td>какой-то текст</td>
<td>какой-то текст</td>
</tr>
</table>
```

## Формы

```
<form action="http://www.example.com/test.asp" method="post/get">
<input type="text" name="lastname" value="Петров" size="30" maxlength="50">
<input type="password">
<input type="checkbox" checked="checked">
<input type="radio" checked="checked">
<input type="submit">
<input type="reset">
```

```
<input type="hidden">

<select>
<option>Яблоки
<option selected>Бананы
<option>Вишня
</select>

<textarea name="Комментарии" rows="60" cols="20"></textarea>

</form>
```

## Символьные объекты

```
&lt; то же самое что <
&gt; то же самое что >
&#169; то же самое что ©
```

## Другие элементы

```
<!-- Это комментарий -->

<blockquote>
Текстовая цитата из некоторого источника.
</blockquote>

<address>
Город <br>
Адрес 1<br>
Адрес 2<br>
</address>
```

## 3.8 HTML и XHTML.

### Отличия xhtml от html

\* Все элементы должны быть закрыты. Теги, которые не имеют закрывающего тега (например, `<img>` или `<br>`) должны иметь на конце `/` (например, `<br />`).

\* Булевы атрибуты записываются в развёрнутой форме. Например, следует писать `<option selected="selected">` или `<td nowrap="nowrap">`.

\* Имена тегов и атрибутов должны быть записаны строчными буквами (например, `<img alt="" />` вместо `<IMG ALT="" />`).

\* XHTML гораздо строже относится к ошибкам в коде; `<` и `&` везде, даже в URL, должны замещаться `&lt;` и `&amp;` соответственно. По рекомендации W3C браузеры, встретив ошибку в XHTML, должны сообщить о ней и не обрабатывать документ. Для HTML браузеры должны были попытаться понять, что хотел сказать автор.

\* Кодировкой по умолчанию является UTF-8 (в отличие от HTML, где кодировкой по умолчанию является ISO 8859-1).

Для XHTML страниц рекомендуется задавать MIME-тип — `application/xhtml+xml`, но это не является обязательным, более того — браузер Internet Explorer 6 и младшие версии, не смогут обрабатывать страницу, поэтому традиционно используется стандартный MIME-тип — `text/html`.

Также стандарт рекомендует указание `<?xml version="1.0" encoding="utf-8"?>` перед DTD, но это не обязательно, более того — браузер Internet Explorer воспринимает такое указание (как и любой другой текст перед `<!DOCTYPE>`), как признак того, что данную страницу необходимо отображать в режиме обратной совместимости, а не согласно стандарту.

### **3.9 Введение в CSS**

#### **CSS: CSS1, CSS 2, CSS2.1, CSS3.0, Поддержка браузерами**

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения и других аспектов представления документа. Основной целью разработки CSS являлось разделение содержимого (написанного на HTML или другом языке разметки) и представления документа (написанного на CSS). Это разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печать, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими Шрифт Брайля.

Стандарт CSS определяет приоритеты, в порядке которых применяются правила стилей, если для какого-то элемента подходят несколько правил одновременно. Это называется «каскадом», в котором для правил рассчитываются приоритеты или «веса», что делает результаты предсказуемыми.

Технология CSS: общий взгляд

Название "Каскадные таблицы стилей" происходит от английского Cascading Style Sheets, аббревиатурой которого является CSS. Впервые стили появляются в HTML 4.0 для определения представления элементов HTML и решения проблем представления документов. Стили обычно хранятся в таблицах стилей: могут быть определены как внутри HTML-документа, так и в специальном файле с расширением `css`. Используя отдельные файлы для хранения таблиц стилей, можно существенно сократить объем работы. Также можно определить несколько стилей, которые, подчиняясь существующим правилам, будут каскадно задавать один определенный стиль.

#### **Пример использования CSS**

С помощью CSS документы HTML можно выводить, используя различные стили вывода.

#### **Предназначение стилей**

Язык HTML был создан для описания содержимого документа. Его теги были предназначены для определения "заголовка", "параграфа", "таблицы" (`<h1>`, `<p>`, `<table>` и т.д.). Первоначально для представления документов не было предусмотрено каких-либо тегов форматирования, т.е. предполагалось, что о представлении документа позаботится браузер.

Но это породило процесс создания своих тегов и атрибутов к исходной спецификации HTML-браузерами Netscape и Internet Explorer (такие, например, как тег `<font>` и атрибут `color`), что затруднило создание Web-сайтов, на которых содержимое документов HTML было четко отделено от уровня представления документа.

В этой ситуации консорциум W3C создала при разработке стандарта HTML 4.0 каскадные таблицы стилей.

Таким образом, необходимость разделения содержимого документа и его представления привела к созданию технологии CSS.

Сегодня эту технологию поддерживают все основные браузеры (Internet Explorer, Mozilla Firefox, Opera и др.).

## **Таблицы стилей могут существенно сократить объем работы**

Таблицы стилей определяют вывод элементов HTML (как, например, тег `font` и атрибут `color` в HTML 3.2). Как уже говорилось, каскадные таблицы стилей обычно хранятся во внешних файлах с расширением `.css`. Этот способ удобен тем, что позволяет изменить внешний вид и компоновку всех необходимых страниц в Web, редактируя только один, единственный документ CSS.

## **Приоритет использования стилей**

**Для каскадных таблиц стилей определен приоритет использования.** Если для элемента HTML определено более одного стиля, то все стили будут последовательно "каскадированы" в новую "виртуальную" таблицу стилей, согласно следующим правилам:

- стили, используемые по умолчанию браузером;
- стили, хранящиеся во внешней таблице;
- стили, хранящиеся во внутренней таблице стилей (внутри тега `<head>`);
- встроенный стиль (внутри элемента HTML).

Эти правила определяют порядок увеличения приоритета стилей.

Таким образом, встроенный стиль (внутри элемента HTML) имеет наивысший приоритет, т.е. будет переопределять стиль, который объявлен в теге `<head>`, во внешней таблице стилей или в браузере (значение по умолчанию).

### **Синтаксис**

Синтаксис CSS состоит из трех частей: селектора, свойства и значения:

```
селектор {свойство: значение}
```

Селектор — элемент/тег HTML, который необходимо определить. Свойство — атрибут, который желательно изменить. Каждое свойство может принимать значение. Существует ряд правил и рекомендаций, которые необходимо запомнить.

- Свойство и значение разделяются двоеточием и помещаются внутри фигурных скобок:

```
p {font-size: 75%}
```

- Если значение состоит из нескольких слов, то необходимо поместить значение в кавычки:

```
h1 {font-family: "lucida calligraphy"}
```

- Если требуется определить более одного свойства, то необходимо разделить свойства точкой с запятой:

```
table { font-family: arial, "sans serif"; border-style: dotted}
```

- Чтобы определения стилей было удобно читать, можно каждое свойство писать на отдельной строке:

```
h2
{
font-family: arial;
margin-right: 20pt;
color:#ffffff }
```

- При определении правил допускается группировка селекторов, при этом в качестве разделителя селекторов используется запятая. В следующем примере в группу были объединены все элементы абзацев, таблиц и списков. Все эти элементы будут выведены шрифтом sans serif:

```
p,table,li
{
font-family: "sans serif";
}
```

## Селектор класса (class)

При Web-разработке часто возникает необходимость в задании различных стилей для одного и того же типа элемента HTML. Для решения этой задачи используют селектор класса.

Предположим, что в документе требуется два типа заголовков: более крупный заголовок должен иметь внешний отступ 10 пунктов, а второй — 20 пунктов. Вот как это можно сделать с помощью стилей:

```
h1.stepleft {margin-left: 10pt}
h2.stepright {margin-left: 20pt}
```

Для применения этих стилей в документе HTML необходимо будет использовать атрибут класс:

```
<h1 class="stepleft">
Заголовок с внешним отступом 10 пунктов.
</h1>

<h2 class="stepright">
```

Заголовок с внешним отступом 20 пунктов.  
</h2>

В тоже время не допускается следующее определение атрибута class:

```
< h1 class="stepleft" h2 class="stepright">
```

т.е. можно определить только один атрибут class

Можно также опустить имя тега в селекторе, чтобы определить стиль, который будет использоваться всеми элементами HTML, имеющими определенный class.

```
.left {margin-left: 40pt}
```

В следующем примере все элементы HTML, имеющие class="left", будут иметь внешний отступ, равный 40 пунктам.

В следующем коде элементы table и p имеют class="left". Это означает, что оба элемента будут следовать правилам в селекторе ".left":

```
<table class="left">
```

Эта таблица будет иметь внешний отступ, равный 40 пунктам.

```
</table>
```

```
<p class="left">
```

Этот параграф будет иметь внешний отступ, равный 40 пунктам.

```
</p>
```

Для корректности обработки кода браузерами не рекомендуется начинать имя класса с числа, т.к. это не будет работать в Mozilla/Firefox.

## Селектор идентификатора (id)

Стили элементов HTML можно определить также с помощью селектора идентификатора, который определяется символом #.

Следующее правило стиля будет применимо к элементу, который имеет атрибут id со значением "fontsz":

```
#fontsz{font-size: 50%}
```

Следующее правило стиля будет применимо к элементу ul, который имеет атрибут id со значением "first":

```
ul#first  
{  
list-style: disc;
```

```
color: #ffffff
}
```

Для корректности обработки кода браузерами не рекомендуется начинать имя ID с числа, т.к. это не будет работать в Mozilla/Firefox.

## Комментарии CSS

Для пояснения кода и для последующего его редактирования используются комментарии, которые игнорируются браузерами. Комментарий CSS начинается символом "/\*" и заканчивается символом "\*/", как в примере ниже:

```
/* Первый комментарий. */
table
{
/* Второй комментарий. */
margin-left: 10pt;;
font-family: "sans serif"
/* Третий комментарий. */
border-style:outset;
}
```

## Параметры фона в CSS: подробное рассмотрение

Как было показано ранее, параметры фона в CSS позволяют управлять цветом фона элемента, задавать в качестве фона изображение, повторять циклически фоновое изображение вертикально или горизонтально и позиционировать изображение на странице.

Данные свойства поддерживаются следующими браузерами (в скобках сокращенный вариант, который применяется далее в таблицах параметров и значений): Internet Explorer (IE), Firefox (F), Netscape(N).

Параметр	Описание	Значения	IE	F	N	W3C
background	Служит для задания всех параметров фона в одном объявлении	background-color	4	1	6	1*
		background-image				
		background-repeat				
		background-attachment				
		background-position				
background-attachment	Задаёт для изображения фиксированное расположение или перемещающееся вместе с остальной страницей	scroll fixed	4	1	6	1

background-color	Задает цвет фона элемента	color-rgb	4 1 4 1
		color-hex	
		color-name	
		transparent	
background-image	Задает в качестве фона изображение	url	4 1 4 1
		none	
background-position	Задает начальное положение фонового изображения	top left	4 1 6 1
		top center	
		top right	
		center left	
		center center	
		center right	
		bottom left	
		bottom center	
		bottom right	
		x-% y-%	
x-pos y-pos			
background-repeat	Определяет, будет ли и каким образом будет повторяться фоновое изображение	repeat	4 1 4 1
		repeat-x	
		repeat-y	
		no-repeat	

(\* здесь и далее число в столбце "W3C" указывает в какой спецификации CSS определен параметр (CSS1 или CSS2)).

### Параметр 'background'

Этот параметр предназначен для задания всех свойств фона в одном объявлении.

**Наследование:** нет.

**Примеры:**

```
body
{
background: url (picture.gif)
}
```

```
body
{
background: url (http://www.intuit.ru/speciality/image.gif) repeat scroll
}
```

```
body
{
background: yellow url (http://www.intuit.ru/speciality/image.gif) repeat-x
bottom
}
```

### Может принимать следующие значения:

Значение	Описание
background-color	В этом объявлении можно задать от одного до пяти свойств фона
background-image	
background-repeat	
background-attachment	
background-position	

Рассмотрим их подробнее.

#### *Параметр 'background-attachment'*

Этот параметр определяет, будет ли фоновое изображение зафиксировано в определенном месте или будет перемещаться вместе со всей страницей.

**Наследование:** нет.

### Может принимать следующие значения:

Значение	Описание
Scroll	Фоновое изображение перемещается, когда перемещается страница
Fixed	Фоновое изображение не перемещается, когда перемещается страница

#### **Пример:**

```
body
{
background-attachment: fixed;
background-repeat: repeat
}
```

#### *Параметр 'background-color'*

Этот параметр задает фоновый цвет элемента.

**Наследование:** нет.

### Может принимать следующие значения:

Значение	Описание
----------	----------

**color** Значение color может быть названием цвета (red), значением rgb (rgb(255,0,0)) или шестнадцатеричным числом (#ff0000)

**transparent** Фонový цвет является прозрачным

#### **Пример:**

```
h1
{
background-color: gray;
font-family: arial
}
```

#### **Параметр 'background-image'**

Данный параметр задает изображение в качестве фона.

**Наследование:** нет.

**Может принимать следующие значения:**

Значение	Описание
url	Путь доступа к изображению
none	Фоновое изображение нет

#### **Пример:**

```
body
{
background-image: url(http://www.intuit.ru/speciality/image.gif);
background-repeat: repeat;
background-attachment: fixed
}
```

В тех случаях, когда изображение недоступно, необходимо задать цвет background-color, который будет использован.

#### **Параметр 'background-position'**

Этот параметр задает начальное положение фонового изображения.

**Наследование:** нет.

**Может принимать следующие значения:**

Значение	Описание
top left top center top right center left center center center right bottom left bottom center bottom right	Если определить только одно ключевое слово, то вторым значением подразумевается "center"
x-% y-%	Первое значение является горизонтальной координатой, второе значение — вертикальной. Верхний левый угол — 0% 0%. Правый нижний угол — 100% 100%. Если определено только одно значение, то вторым значением подразумевается 50%.
x-pos y-pos	Первое значение является горизонтальной координатой, второе значение — вертикальной. Верхний левый угол — 0 0. Единицами измерения

могут быть пиксели (0px 0px) или любые другие единицы измерения CSS. Если определено только одно значение, то вторым значением подразумевается 50%. Можно смешивать % и эти координаты.

#### Пример:

```
body
{
background-image: url(http://www.intuit.ru/speciality/image.gif);
background-position: right top;
background-attachment: fixed
}
```

```
body
{
background-image: url(http://www.intuit.ru/speciality/image.gif);
background-repeat: no-repeat;
background-position: 100% 100%;
background-attachment: fixed
}
```

#### Параметр 'background-repeat'

Этот параметр определяет, каким образом будет повторяться фоновое изображение.

**Наследование:** нет.

**Может принимать следующие значения:**

Значение	Описание
repeat	Фоновое изображение будет повторяться по вертикали и по горизонтали
repeat-x	Фоновое изображение будет повторяться по горизонтали
repeat-y	Фоновое изображение будет повторяться по вертикали
no-repeat	Фоновое изображение будет выведено только один раз

#### Пример:

```
body
{
background-image: url(http://www.intuit.ru/departament/image.gif);
background-repeat: repeat;
background-attachment: fixed
}
```

Параметры текста CSS позволяют управлять внешним видом текста. Можно изменять цвет текста, увеличивать или уменьшать интервал между символами, выравнивать текст, оформлять текст, делать отступ для первой строки текста и т.д.

**Поддержка браузеров:** Internet Explorer, Firefox, Netscape.

Параметр	Описание	Значения	IE	F	N	W3C
color	Задаёт цвет текста	Color	3	1	4	1
direction	Задаёт направление текста	rtl	6	1	6	2
letter-spacing	Увеличивает или уменьшает интервал между символами	length	4	1	6	1

text-align	Выравнивает текст в элементе	inherit	4	1	4	1
text-decoration	Дополнительное оформление текста	blink	4	1	4	1
text-indent	Делает отступ для первой строки текста элемента	%	4	1	4	1
text-shadow		length				
text-transform	Управляет символами элемента	lowercase	4	1	4	1
unicode-bidi		bidi	5			2
white-space	Задаёт способ обращения с пробелами внутри элемента	nowrap	5	1	4	1
word-spacing	Увеличивает или уменьшает пробел между словами	length	6	1	6	1

## Примеры

- Этот пример показывает, как задать цвет текста.

```
<html>

<head>
<style type="text/css">
p {color: green}
ul {color: #dda0dd}
ol {color: rgb(0,0,255)}
</style>
</head>

<body>
<ul>
<li>список ul</li>
</ul>
<ol>
<li>список ol</li>
</ol>
<p>это параграф</p>
</body>

</html>
```

- Этот пример показывает, как задать фоновый цвет части текста.

```
<html>
<head>
<style type="text/css">
span.back
{
background-color: gray
}
</style>
</head>

<body>
<p>
Данный текст содержит определение, фон которого выделен. <span
class="back">Это
определение.</span>
</p>
</body>
```

```
</html>
```

- Данный пример показывает, как увеличить или уменьшить интервал между символами.

```
<html>

<head>
<style type="text/css">
p {letter-spacing: 1cm}
li {letter-spacing: 5px}
</style>
</head>

<body>
<p>параграф</p>
<ol>
<li>элемент списка</li>
</ol>
</body>

</html>
```

- Данный пример показывает, как выравнивать текст.

```
<html>
<head>
<style type="text/css">
ol {text-align: center}
ul {text-align: left}
dl {text-align: right}
</style>
</head>

<body>
<ol>
<li>список ol</li>
<li> список ol</li>
<li> список ol</li>
</ol>
<ul>
<li> список ul</li>
<li> список ul</li>
<li> список ul</li>
</ul>
<dl>
<dt> список <dd>dl dl dl</dd></dt>
<dt> список <dd>dl dl dl</dd></dt>
<dt> список <dd>dl dl dl</dd></dt>
</dl>
</body>
</html>
```

- Этот пример показывает, как можно оформить текст.

```
<html>
<head>
<style type="text/css">
a {text-decoration: underline}
ul {text-decoration: overline}
ol {text-decoration: line-through}
</style>
```

```
</head>

<body>
<ol>
<li>первое</li>
<li>второе</li>
<li>третье</li>
</ol>
<ul>
<li>1</li>
<li>2</li>
<li>3</li>
</ul>
<p><a href="http://www.intuit.ru/">www.intuit.ru</a></p>
</body>

</html>
```

- Этот пример показывает, как сделать отступ для первой строки параграфа.

```
<html>
<head>
</head>

<body>
<p>
параграф<br>
<p style="text-indent: 2cm;">
параграф<br>
<p style="text-indent: 4cm;">
параграф<br>
</p>
</body>

</html>
```

- Данный пример показывает, как управлять регистром символов в тексте.

```
<html>
<head>
</head>

<body>
<pre style="text-transform: uppercase;">Верхний регистр</pre>

<p style="text-transform: lowercase;">Нижний регистр</p>

<pre style="text-transform: capitalize;">первые буквы в словах
заглавные</pre>
</body>

</html>
```

## Параметры текста в CSS: подробное рассмотрение

### *Параметр 'color'*

Этот параметр задает цвет текста.

**Наследование:** да.

**Может принимать следующие значения:**

<b>Значение</b>	<b>Описание</b>
color	Значением color может быть название цвета (red), значение rgb (rgb(255,0,0)) или шестнадцатеричное значение (#ff0000).

## Пример:

```
h1
{
color: green
}
```

### *Параметр 'direction'*

Параметр задает направление текста.

**Наследование:** да.

**Может принимать следующие значения:**

Значение	Описание
ltr	Направление текста слева направо
rtl	Направление текста справа налево

## Пример:

```
p
{
direction: rtl
}
```

### *Параметр 'letter-spacing'*

Данный параметр увеличивает или уменьшает интервал между символами.

**Наследование:** да.

**Примечание:** допускаются отрицательные значения.

**Может принимать следующие значения:**

Значение	Описание
normal	Определяет обычный пробел между символами
Length	Определяет фиксированный пробел между символами

## Примеры:

```
pre
{
```

```
letter-spacing: -2px
}

pre
{
letter-spacing: 20px
}
```

### ***Параметр 'text-align'***

Этот параметр задает выравнивание текста в элементе.

**Наследование:** да.

**Может принимать следующие значения:**

Значение	Описание
left	Выравнивает текст слева
right	Выравнивает текст справа
center	Центрирует текст
justify	

**Пример:**

```
h1
{
text-align: right
}
```

### ***Параметр 'text-decoration'***

Данный параметр задает дополнительное оформление текста.

**Наследование:** нет.

**Примечание:** цвет оформления должен быть задан свойством "color".

**Может принимать следующие значения:**

Значение	Описание
None	Определяет обычный текст
Underline	Определяет линию под текстом
Overline	Определяет линию над текстом
line-through	Определяет линию через текст
Blink	Определяет мигающий текст

## Пример:

```
h1
{
text-decoration: overline
}
```

## *Параметр 'text-indent'*

Данный параметр создает отступ для первой строки текста элемента.

**Наследование:** да.

**Примечание:** допускаются отрицательные значения, для которых первая строка будет сдвинута влево.

**Может принимать следующие значения:**

Значение	Описание
Length	Определяет фиксированный отступ
%	Определяет отступ в % от ширины родительского элемента

## Примеры:

```
pre
{
text-indent: -10px
}

p
{
text-indent: 10px
}
```

## *Параметр 'text-transform'*

Этот параметр управляет регистром символов в элементе.

**Наследование:** да.

**Может принимать следующие значения:**

Значение	Описание
None	Определяет обычный текст с символами нижнего регистра и заглавными буквами

Capitalize Каждое слово в тексте начинается с заглавной буквы  
Uppercase Определяет только заглавные буквы  
Lowercase Определяет только символы нижнего регистра

### Примеры:

```
h1
{
text-transform: capitalize
}
```

```
pre
{
text-transform: lowercase
}
```

### *Параметр 'white-space'*

Параметр задает способ обработки пробелов внутри элемента.

**Наследование:** да.

**Может принимать следующие значения:**

Значение	Описание
normal	Браузер игнорирует пробел
pre	Браузер сохраняет пробел. Действует как тег <pre> в HTML
nowrap	Текст не будет переноситься на другую строку, пока не встретится тег  

### Пример:

```
pre
{
white-space: pre
}
```

### *Параметр 'word-spacing'*

Данный параметр увеличивает или уменьшает пробел между словами.

**Наследование:** да.

**Примечание:** допускаются отрицательные значения.

**Может принимать следующие значения:**

Значение	Описание
----------	----------

normal	Определяет обычный пробел между словами
length	Определяет фиксированный пробел между словами

**Примеры:**

```
pre
{
word-spacing: -10px
}

h1
{
word-spacing: 35px
}
```

– **Литература**

<http://www.citforum.ru/internet/html40/cover.html>

<http://www.w3.org/TR/1998/REC-html40-19980424/>

<http://www.webtag.ru/kuru/kucss.php>

<http://www.w3.org/TR/CSS>

<http://www.intuit.ru/department/internet/html/>

<http://www.intuit.ru/department/internet/csscert/>

<http://www.intuit.ru/department/internet/htmlintro/>

<http://www.intuit.ru/department/internet/vinhtmlcss/>