**Лабораторная работа 1.**

Для выполнения лабораторной работы 1:

1. Прочитайте раздел 1 «Работа с HTML тегами и атрибутами». Выполните задание «Связанные ссылками страницы», которое находится после первого раздела. Дополнительно в одну из страниц добавьте любую картинку.

2. Прочитайте раздел 2 «Работа с HTML таблицами». Выполните задание «Закрепление», которое находится после второго раздела. Таблицу из задания поместите в одну из созданных ранее трех страниц.

Для выполнения лабораторной работы 1 никакого дополнительного программного обеспечения не понадобится. Все что необходимо – это блокнот для создания и редактирования ваших страниц и браузер для просмотра содержимого.

Лабораторная работа рассчитана на одно занятие.

**Основы HTML и CSS**

*Языки****HTML****и****CSS****предназначены для верстки сайтов (верстка - это размещение элементов сайта по нужным местам). Язык****PHP****нужен для программирования сайта (с его помощью можно, к примеру, сделать регистрацию пользователей). Язык****JavaScript****нужен для того, чтобы 'оживить' сайт: к примеру, сделать меняющиеся картинки (слайдер).*

**Язык HTML**

Язык **HTML** - это основа web сайтов, с его помощью создается каркас страницы, которую вы видите в браузере.

Если сравнивать страницу сайта и обычную бумажную книгу, то на сайте, как и в книге, есть абзацы и заголовки. В книге есть название всей книги (по сути самый главный заголовок), есть названия глав, параграфов в этих главах и так далее.

Заголовки, абзацы и другие блоки можно выделить и на странице сайта. Это делается с помощью *HTML тегов*.

**Что такое HTML теги?**

**HTML теги** - это специальные команды для браузера. Они говорят ему, что, к примеру, следует считать заголовком страницы, а что абзацем.

Теги строятся по такому принципу: уголок **<**, потом имя тега, а потом уголок **>**, вот так: **<имя тега>**. Имя тега может состоять из английских букв и цифр. Примеры тегов: **<h1>**, **<p>**, **<b>**.

Теги обычно пишутся парами - открывающий тег и соответствующий ему закрывающий. Разница между открывающим и закрывающим тегами в том, что в закрывающем теге после уголка **<** стоит слеш **/**.

К примеру, **<p>** - так я открыл тег p, а так - **</p>** - я его закрыл. Все, что попадает между открывающим и закрывающим тегами, подпадает под воздействие нашего тега.

Бывают теги, которые не нужно закрывать, например, **<br>** или **<img>**.

**Атрибуты**

В тегах также могут размещаться **атрибуты** - специальные команды, которые расширяют действие тега.

Атрибуты размещаются внутри открывающего тега в таком формате: **<тег атрибут1="значение" атрибут2="значение">**.

Кавычки могут быть любыми - одинарными или двойными, допустимо их вообще их не ставить, если значение атрибута состоит из одного слова (но это не желательно).

**Язык CSS**

Язык **CSS** расширяет возможности языка HTML. Он позволяет менять цвета, шрифты, фон, в общем заниматься красотой сайта. А HTML, соответственно, отвечает за структуру сайта.

**1. Работа с HTML тегами и атрибутами**

## 1. Структура простейшей страницы

Страница сайта - это обычный текстовый файл с расширением **.html**. Внутри этого файла и хранится текст HTML страницы вместе с тегами. Этот файл обязательно должен иметь следующие теги: тег **<html>**, который должен содержать в себе текст всего сайта (все, что написано вне этого тега, браузером будет проигнорировано), а внутри него должны быть еще два тега: тег **<head>** для служебного содержимого страницы и тег **<body>** - для основного текста, который и виден на экране браузера.

В служебное содержимое, которое располагается внутри тега **<head>**, входит много различных вещей, но пока нам нужны только две из них. Это тег **<title>**, задающий название страницы, которое будет видно во вкладке браузера, и тег **<meta>**, который задает кодировку страницы (она ставится в атрибуте **charset** и обычно имеет значение **utf-8**).

Кроме того, перед тегом **<html>** обычно пишется конструкция **doctype**, которая указывает версию языка HTML, на которой сделан сайт. Актуальная сейчас версия языка имеет номер пять и доктайп для нее должен выглядеть так - **<!DOCTYPE html>**.

Итак, давайте посмотрим на основную структуру страницы (чтобы запустить это пример в браузере - скопируйте его в текстовый файл с расширением **.html** и откройте в браузере):

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

Это основное содержимое страницы.

</body>

</html>

Как выглядит этот пример в браузере смотрите по [данной ссылке](http://old.code.mu/examples/html/html/index.html).

Думаю, после того, как вы прочитали про основную структуру страницы, у вас по-прежнему остается некоторое недопонимание, как все это выглядит на практике. Поэтому я снял специальное видео, в котором покажу вам, как сделать вашу первую HTML страницу и запустить ее в браузере (в нем я также расскажу про title страницы, про кодировки, про оформление кода). Посмотрите его и только потом переходите к дальнейшему чтению:

Тут скоро будет видео как начать: как сохранить с расширением .html, структура страницы, кодировки.

Ну, а теперь, научившись создавать простейшие страницы, мы перейдем к изучению полезных тегов, которые следует использовать внутри тега **<body>**. Это будут теги для абзацев, заголовков, списков, ссылок и других полезных вещей. Итак, приступим.

## 2. Абзацы

Одним из основных элементов страницы являются **абзацы**. Их можно сравнить с абзацами в книге - каждый абзац начинается с новой строки и имеет так называемую красную строку (это когда первая строка текста абзаца немного отступает вправо). По умолчанию красной строки нет, но ее легко сделать (об этом чуть позже).

Абзац создается с помощью тега **<p>** таким образом:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<p>

Это абзац.

</p>

<p>

Это еще один абзац.

</p>

<p>

И еще один абзац.

</p>

</body>

</html>

Так код будет выглядеть в браузере:

Это абзац.

Это еще один абзац.

И еще один абзац.

## 3. Заголовки h1, h2, h3, h4, h5, h6

Кроме абзацев важное значение на странице имеют **заголовки**. Их также можно сравнить с заголовками из книги - каждая глава имеет свой заголовок (название этой главы) и разбита на параграфы, которые тоже имеют свои заголовки. Ну, а основной текст страницы располагается в абзацах.

**Заголовки создаются** с помощью тегов **<h1>**, **<h2>**, **<h3>**, **<h4>**, **<h5>**, **<h6>**. Они имеют разную степень важности. В заголовке **h1** следует располагать **название всей HTML страницы**, в **h2** - название **блоков** страницы, в **h3** - название подблоков и так далее.

Все заголовки по умолчанию жирные и имеют разный размер (это можно поменять через CSS, но об этом позже). Смотрите пример:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<h1>Заголовок h1</h1>

<h2>Заголовок h2</h2>

<h3>Заголовок h3</h3>

<h4>Заголовок h4</h4>

<h5>Заголовок h5</h5>

<h6>Заголовок h6</h6>

<p>Это первый абзац.</p>

<p>Это второй абзац.</p>

<p>Это третий абзац.</p>

</body>

</html>

Так код будет выглядеть в браузере:

# Заголовок h1

## Заголовок h2

### Заголовок h3

#### Заголовок h4

##### Заголовок h5

###### Заголовок h6

Это первый абзац.

Это второй абзац.

Это третий абзац.

## 4. Жирный

Вы уже знаете, что заголовки по умолчанию **жирные**. Однако, можно сделать жирным и обычный текст - достаточно взять его в тег **<b>**. Смотрите пример:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<p>

Это обычный текст, а это <b>жирный</b> текст.

</p>

</body>

</html>

Так код будет выглядеть в браузере:

Это обычный текст, а это **жирный** текст.

Обратите внимание на то, что тег **<b>** следует использовать внутри какого-либо другого тега, например абзаца. В этом случае абзацы создают общую структуру страницы (абзацы и заголовки), а тег **b** делает жирным отдельные кусочки текста.

## 5. Курсив

Кроме жирного можно сделать также и *курсив* с помощью тега **<i>**:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<p>

Это обычный текст, а это <i>курсивный</i> текст.

</p>

</body>

</html>

Так код будет выглядеть в браузере:

Это обычный текст, а это *курсивный* текст.

## 6. Списки

Наряду с абзацами и заголовками существует еще один важный элемент страницы - это **списки**. Такие элементы наверняка знакомы всем пользователям интернета. Они представляют собой перечисление чего-либо (список) по пунктам. Возле каждого пункта списка обычно стоит закрашенный кружок (он называется **маркер** списка).

Списки создаются с помощью тега **<ul>**, внутри которого обязательно должны идти теги **<li>**. Тег **ul** задает сам список, а в теги **li** следует ложить пункты списка (то есть одному li соответствует один маркер списка). Имейте ввиду, что отдельно теги li использовать нельзя. Смотрите пример:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<ul>

<li>Первый пункт списка.</li>

<li>Второй пункт списка.</li>

<li>Третий пункт списка.</li>

</ul>

</body>

</html>

Так код будет выглядеть в браузере:

* Первый пункт списка.
* Второй пункт списка.
* Третий пункт списка.

Списки, созданные через тег **ul**, называются **неупорядоченными** списками. Такое название им дано потому, что существуют еще и **упорядоченные** списки, у которых вместо маркеров-точек будут маркеры-цифры. Такие списки вместо тега ul имеют тег **<ol>**, а пункты таких списков также создаются через теги **li**.

Давайте сделаем упорядоченный список, используя тег **ol**:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<ol>

<li>Первый пункт списка.</li>

<li>Второй пункт списка.</li>

<li>Третий пункт списка.</li>

</ol>

</body>

</html>

Так код будет выглядеть в браузере:

1. Первый пункт списка.
2. Второй пункт списка.
3. Третий пункт списка.

Удобство упорядоченных списков в том, что я могу вставить новый пункт списка в любое место - и нумерация сама перестроится (то есть мне не приходится за ней следить в случае каких-либо изменений, как было бы в том случае, если бы я расставлял ее вручную).

## 7. Ссылки

**Ссылки** являются теми элементами, которые делают из интернета интернет. Нажимая на ссылки, мы можем переходить с одной страницы сайта на другу. Если бы их не было - интернет был бы просто набором страниц, никак не связанных друг с другом.

Ссылка создается с помощью тега **<a>**, при этом у нее должен быть обязательный атрибут **href**, в котором хранится адрес той страницы, на которую ведет ссылка. Смотрите пример:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<a href="https://tpu.ru">Ссылка на сайт tpu.ru.</a>

</body>

</html>

Так код будет выглядеть в браузере:

[Ссылка на сайт tpu.ru.](https://tpu.ru/)

Ссылки бывают **абсолютные** и **относительные**, кроме того, они могут вести как на ваш сайт, так и на чужой. Эти сложные моменты лучше показывать, а не описывать текстом, поэтому я снял для вас следующее видео. Посмотрите его и только потом переходите к дальнейшему чтению:

## 8. Картинки

Давайте теперь разберемся с тем, как разместить **изображение** на странице вашего сайта. Для этого предназначен тег **<img>**, имеющий обязательный атрибут **src**, в котором хранится путь к файлу картинки.

Как это работает, посмотрим на следующем примере:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<img src="smile.png">

</body>

</html>

Так код будет выглядеть в браузере:



Обратите внимание на то, что тег **<img>** не требует закрывающего тега.

## 9. Ссылки в виде картинок

**Ссылкой** может быть не только текст, но и **картинка** - для этого достаточно тег **<img>** вложить в тег **<a>**, как это сделано в следующем примере. Нажмите на картинку - и вы перейдете по ссылке на сайт phphtml.net (чтобы потом вернуться назад на книжку - нажмите кнопку "назад" в браузере):

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<a href="https://tpu.ru"><img src=" smile.png"></a>

</body>

</html>

Так код будет выглядеть в браузере:

[](https://tpu.ru/)

## 10. Разрыв строки

Вспомните, что будет, если сделать, к примеру, два абзаца рядом - в этом случае текст, который лежит в каждом из них, будет начинаться с **новой** строки.

Давайте убедимся в этом на следующем примере:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<p>Это первый абзац.</p>

<p>Это второй абзац.</p>

</body>

</html>

Так код будет выглядеть в браузере:

Это первый абзац.

Это второй абзац.

Бывают, однако, ситуации, когда мы хотели бы, чтобы у нас был один абзац, но некоторый текст в нем начинался с новой строки. Зачем такое может понадобится? К примеру, я хочу набрать стихотворение, но не хочу разбивать каждую его строку в отдельный абзац, так как это было бы не очень логично.

Чтобы сделать такое, в том месте, где должен быть разрыв строки, следует написать тег **<br>**. Учтите, что этот тег особенный и не имеет закрывающего.

Давайте посмотрим на его работу на следующем примере:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<p>Это первая строка текста, <br> а это вторая.</p>

</body>

</html>

Так код будет выглядеть в браузере:

Это первая строка текста,  
а это вторая.

## 11. Комментарии HTML

Практически во всех языках программирования есть такое понятие как "комментарии". **Комментарии** - это такой текст, который будет проигнорирован браузером - на экране он будет не виден, но останется в исходном коде страницы.

Для чего нужны комментарии? **Во-первых**, для того, чтобы оставлять в исходном коде пометки себе или тому человеку, который будет править ваш код после вас. **Во-вторых**, с помощью комментариев можно убирать некоторые вещи, которые вы бы сейчас хотели удалить, но потом, может быть, захотите вернуть. Их можно просто временно *закомментировать*, а потом *откомментировать* при необходимости.

Комментарии в HTML оформляются следующим образом: сначала уголок, знак ! и два дефиса - **<!--**, потом текст комментария, а потом два дефиса и уголок - **-->**. Смотрите пример:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Это заголовок</title>

</head>

<body>

<!--

Это комментарий HTML -

на странице сайта он не отображается.

-->

Это основное содержимое страницы.

</body>

</html>

**Задания**

### Заголовки, жирность, курсив

**1. Повторите страницу по данному по образцу:**

# Это заголовок

## Это заголовок

### Это заголовок

#### Это заголовок

Это **абзац**.

Это еще *абзац*.

# *Это заголовок h1*

**2. Повторите страницу по данному по образцу:**

# Что такое CMS

**CMS** - «система управления контентом» (**движок**) – написанная PHP-программистами основа для сайта, с помощью которой вы сможете управлять сайтом (добавлять контент, менять пункты меню и т.п.) не зная HTML и CSS.

Однако, для того чтобы сделать сайт с помощью **CMS** *потребуются услуги* и программиста, и дизайнера, и верстальщика. И капиталовложения.

## Какие бывают cms

Бывают различные системы управления контентом: для интернет-магазинов, для блогов, для форумов и т.д.

## Примеры cms

*Примеры популярных CMS*: Joomla, WordPress (для блогов), PhpBB (для форумов).

**CMS-ки** бывают *платные* и *бесплатные*.

### Списки

**3. Повторите страницу по данному по образцу:**

# Списки

## Список цветов

* Красный
* Желтый
* Зеленый
* Синий

## Список студентов

1. Иванов
2. Петров
3. Сидоров
4. Николаев

## Список студентов

1. Иванов
   * Возраст - 23 года
   * Курс - 3
2. Петров
   * Возраст - 19 лет
   * Курс - 2
3. Сидоров
   * Возраст - 18 лет
   * Курс - 1

### Заголовки, абзацы и списки

**4. Повторите страницу по данному по образцу:**

# Что нужно знать, чтобы делать сайты

1. **HTML**
2. *CSS*
3. PHP
4. SQL
5. JavaScript
6. jQuery
7. Flash
8. SEO

## PHP и JavaScript

Языки программирования **PHP** и **JavaScript** позволяют сделать сайт динамичным, то есть реагирующим на действия пользователя. Например, можно сделать красивую выпадающую менюшку или слайдер

## Виды скриптов

Для этого пишутся скрипты (англ. *script* - «сценарий») - программы, позволяющиее реагировать на действия пользователя. Скрипты бывают двух видов:

* те, которые выполняются на сервере, а результат их выполнения приходит в браузер к пользователю уже в готовом виде. Это скрипты, написанные на языке **PHP**. На нем пишуться **CMS-ки** – системы управления контентом.
* те, которые выполняются прямо в браузере пользователя. Это скрипты, написанные на языке **JavaScript**. Они чаще всего используются для, того чтобы сделать страницу более удобной и красивой.

### Связанные ссылками страницы

**5. Повторите 3 страницы и свяжите их ссылками друг с другом:**

**страница 1:**

# SEO

Здесь вставить ссылку на вторую страницу

## Что такое seo

**SEO** (СЕО) - комплекс мер по продвижению сайта в поисковых системах и выводу положения сайта в **«топ»** - то есть так, чтобы сайт по интересующему нас запросу находился на первой странице поисковика (а туда, как известно, помещается только 10 ссылок).

Это нужно для увеличения посещаемости сайта, так как большая часть посетителей сайта приходит именно с поисковых систем. Поэтому, *чем выше наш сайт в выдаче поисковика – тем больше у него посетителей*.

## Популярные поисковики

Самыми популярными поисковиками на нашей территории являются *Google и Yandex* и именно за их внимание к своему сайту и ведется постоянная борьба между веб-мастерами. Позиции сайта в выдаче поисковика зависят от **релевантности** (соответствия) запроса пользователя содержимому сайта и от специальных индексов. Для гугла это **PR** (Page Rank), а для Яндекса – **ТИЦ** (*тематический индекс цитирования*). Чем больше эти показатели, тем выше будут позиции сайта по сравнению с конкурентами.

Пусть, например, у нас есть сайт, посвященный **городу Минску** с названием «город Минск» с ТИЦ 100 и PR 2. И есть сайт- конкурент с названием «Все о городе Минске» с ТИЦ 200 и PR 1. Тогда по запросу «сайт о городе Минск» позиции нашего сайта будут ниже, чем у конкурента, а в гугле наоборот наши будут выше (т.к. наш ТИЦ меньше ТИЦ конкурента, а с PR наоборот – наш больше).

**страница 2:**

# Списки

Здесь вставить ссылку на третью страницу

## Список языков

* PHP
* HTML
* CSS
* JavaScript

## Список CMS

1. joomla
2. wordpress
3. netcat
4. phpBB
5. modx
6. bitrix

**страница 3:**

# ТИЦ и PR

Здесь вставить ссылку на первую страницу

## От чего зависит ТИЦ и PR

**ТИЦ** и **PR** зависят от многих вещей и точный алгоритм их определения держиться в тайне. Больше всего на них влияет **количество и вес ссылок** на сайт с других сайтов. Причем чем больше ТИЦ и PR ссылающихся сайтов, тем *больше и вес*, который передает ссылка. Также вес зависит от того, насколько тематичными являются ссылающиеся сайты (то есть на сколько их тема соответсвует нашей).

Итак, подведем итог. **Тиц и PR зависят от**

* *количества ссылок*
* *веса ссылок*
* *возраста сайта*

## Меры по продвижению сайтов

Основными мерами по продвижению сайта в поисковиках являются

1. получение тематических ссылок на свой сайт с других сайтов
2. внутренняя оптимизация сайта

### Картинки

**6. Повторите страницу по данному по образцу:**



**2. Работа с HTML таблицами**

**1. Структура простейшей таблицы**

Что такое таблица в обычной жизни - это набор **строк** и **столбцов**, на пересечении которых находятся **ячейки**.

В **HTML** таблицы создаются по похожему принципу. Там тоже есть столбцы и строки с ячейками, однако HTML код таблиц при первом взгляде может показаться непривычным.

Сама таблица имеет жесткую структуру: главным является тег **<table>**, внутри которого должны лежать теги **<tr>**, которые создают **ряды** (строки) таблицы, а внутри них - теги **<td>**, которые создают **ячейки**.

Как вы видите, нету тегов, которые создавали бы **столбцы** - таблицы создаются по рядам: сначала первый ряд, потом второй и так далее.

Изучите внимательно следующий пример с таблицей (тегу **table** добавлен атрибут **border**, который задает границу таблице и ее ячейкам):

<table border="1">

<!--Это будет первый ряд таблицы:-->

<tr>

<td>Ячейка 1</td>

<td>Ячейка 2</td>

<td>Ячейка 3</td>

</tr>

<!--Это будет второй ряд таблицы:-->

<tr>

<td>Ячейка 4</td>

<td>Ячейка 5</td>

<td>Ячейка 6</td>

</tr>

<!--Это будет третий ряд таблицы:-->

<tr>

<td>Ячейка 7</td>

<td>Ячейка 8</td>

<td>Ячейка 9</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| Ячейка 1 | Ячейка 2 | Ячейка 3 |
| Ячейка 4 | Ячейка 5 | Ячейка 6 |
| Ячейка 7 | Ячейка 8 | Ячейка 9 |

**2. Ячейки-заголовки**

Кроме тегов **td** существуют также теги **<th>**, которые также создают ячейки. Но это уже будут не обычные ячейки, а **ячейки-заголовки**, которые указывают, что находится в данном столбце (или строке) таблицы.

В следующем примере ячейки "Иван" и "Николай" должны быть обычными ячейками **td**, а ячейка "Имя" по логике должна быть ячейкой-заголовком **th**, так как "Имя" - это общее название содержимого этого столбца:

<table border="1">

<tr>

<th>Имя</th>

<th>Фамилия</th>

<th>Зарплата</th>

</tr>

<tr>

<td>Иван</td>

<td>Иванов</td>

<td>200$</td>

</tr>

<tr>

<td>Николай</td>

<td>Сидоров</td>

<td>1000$</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

По умолчанию текст в ячейках **th** будет **жирный** и расположен **по центру** (это поведение можно поменять, но об этом позже).

**3. Атрибут cellspacing**

Вы обратили внимание на то, что между ячейками есть некоторое **расстояние**, из-за которого видно то, что каждая ячейка имеет свою границу? Это расстояние регулируется атрибутом **cellspacing**. При этом, если мы не напишем этот атрибут, то его значение будет совсем **не ноль** (в этом случае все ячейки слиплись бы друг с другом). То есть, атрибут **cellspacing** по умолчанию имеет некоторое **ненулевое** значение, из-за которого ячейки не слипаются (это расстояние равно нескольким пикселям).

Давайте посмотрим на примере. Сейчас я увеличу отступ между ячейками до **10** пикселей:

<table border="1" cellspacing="10">

<tr>

<th>Имя</th>

<th>Фамилия</th>

<th>Зарплата</th>

</tr>

<tr>

<td>Иван</td>

<td>Иванов</td>

<td>200$</td>

</tr>

<tr>

<td>Николай</td>

<td>Сидоров</td>

<td>1000$</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

То, что **cellspacing** имеет некоторое значение по умолчанию, зачастую может мешать вам. В этом случае следует просто поставить его в ноль.

В HTML5 этот атрибут считается устаревшим (но он по-прежнему придает отступ по умолчанию). Вместо него следует использовать специальное CSS свойство, о котором мы поговорим попозже.

Если вы думаете, что вам не стоит изучать этот атрибут из-за того, что он устарел - вы ошибаетесь. В мире гораздо больше сайтов, сделанных не на HTML5, и в них вы можете столкнуться с этим атрибутом (и некоторыми другими устаревшими, которые мы будем проходить). В этом случае вам нужно знать, что это такое и как с ним работать.

**4. Атрибут cellpadding**

Атрибут **cellpadding** задает отступ между текстом и границей ячейки. В следующем примере я поставлю значение этого атрибута в 20px и теперь текст отойдет от границы ячеек:

<table border="1" cellpadding="20">

<tr>

<th>Имя</th>

<th>Фамилия</th>

<th>Зарплата</th>

</tr>

<tr>

<td>Иван</td>

<td>Иванов</td>

<td>200$</td>

</tr>

<tr>

<td>Николай</td>

<td>Сидоров</td>

<td>1000$</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

Для сравнения привожу то, как таблица будет выглядеть по умолчанию:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

Атрибут **cellpadding** также имеет некоторое значение по умолчанию. Если оно вам мешает - обнулите его.

Этот атрибут, так же, как и cellspacing, считается устаревшим в HTML5.

**5. Обнуляем cellpadding и cellspacing**

Давайте обнулим **cellpadding** и **cellspacing** и посмотрим, что станет с таблицей в этом случае:

<table border="1" cellpadding="0" cellspacing="0">

<tr>

<th>Имя</th>

<th>Фамилия</th>

<th>Зарплата</th>

</tr>

<tr>

<td>Иван</td>

<td>Иванов</td>

<td>200$</td>

</tr>

<tr>

<td>Николай</td>

<td>Сидоров</td>

<td>1000$</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

Как вы видите, в данном случае ячейки таблицы **слиплись**, однако все равно заметно, что каждая ячейка имеет свою границу (то есть границы сейчас двойные). Вы сможете поправить это, когда изучите язык CSS.

**6. Добавляем ширину и высоту**

Существуют также атрибуты **width** и **height**, которые позволяют задать таблице **ширину** и **высоту** таблицы соответственно (без этих атрибутов ширина и высота таблицы регулируются ее содержимым: много текста - ширина большая, мало текста - ширина маленькая).

Посмотрите на пример их применения:

<table border="1" cellpadding="0" cellspacing="0" width="300" height="300">

<tr>

<th>Имя</th>

<th>Фамилия</th>

<th>Зарплата</th>

</tr>

<tr>

<td>Иван</td>

<td>Иванов</td>

<td>200$</td>

</tr>

<tr>

<td>Николай</td>

<td>Сидоров</td>

<td>1000$</td>

</tr>

</table>

Так код будет выглядеть в браузере:

|  |  |  |
| --- | --- | --- |
| **Имя** | **Фамилия** | **Зарплата** |
| Иван | Иванов | 200$ |
| Николай | Сидоров | 1000$ |

Значениями атрибутов могут выступать **пиксели** или **проценты**. Значения в процентах задаются таким образом: **width="30%"** - в этом случае таблица займет **30%** ширины родителя.

**Задания**

### Простые таблицы

**1. Повторите страницу по данному по образцу:**

|  |  |  |
| --- | --- | --- |
| ячейка 1 | ячейка 2 | ячейка 3 |
| ячейка 4 | ячейка 5 | ячейка 6 |
| ячейка 7 | ячейка 8 | ячейка 9 |

### Добавим заголовки, атрибуты width (400) и height (200)

**2. Повторите страницу по данному по образцу:**

|  |  |  |
| --- | --- | --- |
| **ячейка 1** | **ячейка 2** | **ячейка 3** |
| ячейка 4 | ячейка 5 | ячейка 6 |
| ячейка 7 | ячейка 8 | ячейка 9 |

### Добавим выравнивание align

**3. Повторите страницу по данному по образцу:**

|  |  |  |
| --- | --- | --- |
| **ячейка 1** | **ячейка 2** | **ячейка 3** |
| ячейка 4 | ячейка 5 | ячейка 6 |
| ячейка 7 | ячейка 8 | ячейка 9 |

### Закрепление

**4. Повторите страницу по данному по образцу:**

|  |  |  |
| --- | --- | --- |
| **номер** | **тег** | **значение** |
| 1 | p | абзац |
| 2 | a | ссылка |
| 3 | b | жирный текст |
| 4 | i | курсив |
| 5 | table | таблица |
| 6 | tr | ряд таблицы |
| 7 | td | ячейка таблицы |
| 8 | th | заголовок таблицы |