Лекция 1. Введение и основные понятия. Структура гипертекстового документа

Оглавление

1.1. Введение

Содержание курса: фронтэнд-разработка клиентских web-приложений:

- HTML5 + CSS3 + JavaScript (осенний семестр);

Часы:

- осень 34 / 34 / экзамен

Что нужно для работы:

- Notepad++ или другой текстовый редактор с поддержкой кодировки Юникода UTF-8 и подсветкой синтаксиса;
- Актуальные браузеры, на совместимость с которыми нужно проверять свои работы:
  - Internet Explorer 11 или MS Edge;
  - Текущие версии Google Chrome, Mozilla Firefox, возможно, Opera и Android Browser

Что читать по теме:

- Рекомендуемые книги из папки курса (или другие по HTML5, CSS3 и JS):
  - Нидерст Роббинс. HTML5, CSS3 и JavaScript. Исчерпывающее руководство (2014)
  - Б. Лоусон, Р. Шарп. Изучаем HTML 5 (Библиотека специалиста) (2011)
  - Мак-Дональд Мэтью. HTML5. Недостающее руководство (2012)
  - Бен Фрэйн. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств (2017)
- http://webref.ru/ - основной справочник по тегам и стилям
- http://javascript.ru/ - справочник по JS
- http://xiper.net/ - профессиональная front-end разработка
- http://psd-html-css.ru/ - уроки HTML5 и вёрстки
- https://htmlacademy.ru – бесплатный онлайн-курс для начинающих

Основные понятия, *которые Вам уже знакомы*:

- **HTML** (HyperText Markup Language, язык разметки гипертекста) – основной формат разметки документов интернета;

Что означают эти слова?

- "*гипертекст*" – документ содержит ссылки на другие свои части или внешние документы и позволяет не только последовательное прочтение;
- "*язык разметки*" – HTML позволяет встраивать в текст дополнительную информацию о структуре документа, расположении подключаемых к странице файлов, внешнем виде тех или иных частей документа и т.п. Слово "language" в данном случае не передает суть дела. Нужно понимать, что HTML - не язык программирования, а формат для представления данных. Программируемой страница становится при добавлении к ней кода на JavaScript.

Файл в формате HTML имеет тип `.html` или (*устарело*) `.htm`. Он текстовый и размечен текстовыми же командами, называемыми **тегами**. Файл HTML может быть в разных кодировках, но рекомендуется для новых разработок **всегда** использовать кодировку Юникода UTF-8 (*она же кодовая страница 65001*).

- **Web-страница** – результат интерпретации документа HTML, отображаемый в окне браузера. В отличие от *HTML-документа*, может содержать не только текст, но и мультимедийное содержимое, интерактивные программы-скрипты и т.п. Кроме того, web-страница может быть результатом интерпретации не одного, а нескольких документов HTML или же формироваться в результате выполнения программы, работающей на стороне сервера.
- **Web-сайт** – совокупность совместно работающих Web-страниц, связанных общей тематикой и структурой.
- **URL-адрес** - *Uniform Resource Locator* — унифицированный указатель ресурса, система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса (файла). Повторите, из каких частей может состоять URL: https://ru.wikipedia.org/wiki/URL
- **HTTP** (Hypertext Transfer Protocol, "протокол передачи гипертекста") – прикладной протокол, по которому браузер обращается к web-странице, расположенной на удалённом или локальном **хосте** (*сервере*). Сервер с помощью системы **DNS** (*Domain Name System*, "система доменных имён") получает информацию с нужного IP-адреса и возвращает заголовки HTTP + разметку HTML, которую мы видим в браузере. Защищённая версия протокола называется HTTPS. Именно с указания протокола `http://` или `https://` начинаются адреса сайтов и страниц в сети. Современные браузеры обычно скрывают эту часть адреса, но она копируется из адресной строки вместе с полным URL.
- При открытии файла HTML из файловой системы компьютера используется прикладной протокол `file://` , если сайт состоит только из HTML, CSS и JavaScript-кода, он будет работать и по протоколу `file://`

Актуальные стандарты:

- **HTML5**: https://ru.wikipedia.org/wiki/HTML5 ; Технология-конкурент - XHTML, её разработка прекращена в 2010 году https://ru.wikipedia.org/wiki/XHTML
- **CSS3**: https://ru.wikipedia.org/wiki/CSS
- **JavaScript**: https://ru.wikipedia.org/wiki/JavaScript - любая версия от 1.6 - 1.8, поддерживаемая актуальными браузерами.

1.2. Теги и атрибуты

В HTML символы, заключенные в угловые скобки < и >, называются *тегами* (реже - *дескрипторами*).

Начальный или *открывающий* тег указывает, что начинается действие команды разметки, его имя записывается внутри треугольных скобок без дополнительных разделителей: `<p>`, `<div>`, `<span>`.

Конечный, или *закрывающий,* тег (тот, который идет после размечаемого содержимого) содержит символ обратной косой черты после первой угловой скобки.

Например, — конечный тег.
Группа из двух тегов (начального и конечного) вместе с заключенным между ними содержимым называется *элементом.* Например, выделим текст курсивом с помощью элемента <i> языка HTML:

```
<i>Текст</i>
```

Некоторые элементы не имеют содержимого и поэтому не нуждаются в конечном теге, например, тег переноса строки <br>.
Обычно элементы организуются в виде иерархической структуры (в которой одни элементы вложены в другие). Например:

```
<p><i>Текст</i></p>
```

При вложении тегов необходимо соблюдать последовательность их закрытия. Например, такой код использовать нельзя:

```
<p><i>Текст</p></i>
```

Во многих случаях браузер проигнорирует такого рода ошибки, но они всё равно могут привести к неправильной интерпретации документа.
В HTML5 наименования тегов и их параметров рекомендуется набирать символами нижнего регистра.

Для каждого элемента разметки (абзац, разрыв строки, рисунок, ячейка таблицы и т.п.) есть свой тег.
Например, абзацы добавляются с помощью тега , а элементы списка с помощью тега <li>.
При этом многие теги правильно работают только внутри других тегов, например, "ячейка таблицы" внутри "строки таблицы".
В парные теги можно вкладывать другие теги. Например, для простого списка имеем:

```
<ul>
  <li>Элемент списка</li>
</ul>
```

Как уже отмечено, у вложенных тегов всегда нужно следить за правильным порядком закрытия. Вложенный тег не может закрываться позже родительского:

```
<ul><li>Элемент списка</ul></li> <!-- Плохо  -->
<ul><li>Элемент списка</li></ul> <!-- Хорошо -->
```

Не все теги можно вкладывать в другие теги, например тег заголовка <h1> не стоит вкладывать в тег абзаца . Эти правила вложенности для каждого тега вы узнаете постепенно по ходу изучения HTML.

В HTML4 теги делились на **блочные**, по умолчанию занимающие всю доступную ширину страницы и автоматически добавляющие разрывы строки, и **строчные,** предназначенные для разметки содержимого.
В отличие от блочных элементов, браузер не добавляет разрыв строки до и после строчного элемента, поэтому если несколько строчных элементов идут подряд друг за другом, они располагаются на одной строке и переносятся на другую строку при необходимости. В большинстве случаев внутрь строчных элементов допустимо помещать другие строчные элементы, вставлять блочные элементы внутрь строчных запрещено.

В HTML5 эти понятия заменены более сложным набором категорий контента, согласно которым каждый HTML-элемент должен следовать правилам, определяющим, какой контент для него допустим.

Большинство тегов имеют *атрибуты* (*параметры*), предназначенные для передачи информации. Атрибуты указываются после имени тега через пробел в формате `имя="значение"`. То есть, значения заключаются в кавычки, а имена атрибутов в кавычки не заключаются. Есть и атрибуты, состоящие только из имени, тогда часть `="значение"` не указывается.
Если у тега несколько атрибутов, то они перечисляются через пробел. Например:
```
<canvas id="example" width="400" height="400">
```
В этом примере параметру `id` тега `<canvas>` присвоено значение `example`, а параметрам `width` и `height` — значение `400`.

1.3. Глобальные атрибуты

В HTML5 есть набор **глобальных атрибутов**, которые применимы к любому элементу HTML5:

- **accesskey**: определяет клавишу для быстрого доступа к элементу
- **class**: задает класс CSS, который будет применяться к элементу
- **contenteditable**: определяет, можно ли редактировать содержимое элемента
- **contextmenu**: определяет контекстное меню для элемента, которое отображается при нажатии на элемент правой кнопкой мыши
- **dir**: устанавливает направление текста в элементе
- **draggable**: определяет, можно ли перетаскивать элемент
- **dropzone**: определяет, можно ли копировать переносимые данные при переносе на элемент
- **hidden**: скрывает элемент
- **id**: уникальный идентификатор элемента. На web-странице элементы не должны иметь повторяющихся идентификаторов
- **lang**: определяет язык элемента
- **spellcheck**: указывает, будет ли для данного элемента использоваться проверка правописания
- **style**: задает стиль элемента
- **tabindex**: определяет порядок, в котором по элементам можно переключаться с помощью клавиши TAB
- **title**: устанавливает дополнительное описание для элемента
- **translate**: определяет, должно ли переводиться содержимое элемента

Как правило, из всего этого списка наиболее часто используются **class**, **title**, **id** и **style**.

1.4. Пользовательские атрибуты

В HTML5 были добавлены пользовательские атрибуты (custom attributes). Теперь разработчик или создатель веб-страницы сам может определить любой атрибут, предваряя его префиксом *data-*. Например:

```
<input type="button" value="Нажать" data-color="red" >
```
Здесь определен атрибут `data-color`, который имеет значение `"red"`. Хотя для этого элемента, ни в целом в HTML не существует подобного атрибута. Мы его определяем сами и устанавливаем у него любое значение. Обрабатывать эти атрибуты можно с помощью кода на JavaScript.

1.5. Одинарные или двойные кавычки? Строчные или прописные буквы?

Нередко можно встретить случаи, когда в HTML при определении значений атрибутов применяются как одинарные, так и двойные кавычки. Например:
```
<input type='button' value='Нажать'>
```
И одинарные, и двойные кавычки допустимы, хотя чаще применяются двойные. Если же само значение атрибута может содержать двойные кавычки, и в этом случае все значение лучше поместить в одинарные:
```
<input type="button" value='Кнопка "Привет, мир"'>
```
или же заменить двойные кавычки их HTML-сущностью (см. п. 1.12):
```
<input type="button" value="Кнопка &quot;Привет, мир&quot;">
```

Имена тегов и атрибутов можно писать как маленькими, так и большими буквами, но лучше придерживаться единообразия и использовать большие буквы только там, где они нужны.

1.6. Общая структура документа

Весь текст HTML-документа расположен между тегами `<html>` и `</html>`. В теге `html` желательно установить атрибут `lang`, значением которого будет код основного языка документа.
Только тег `<!doctype>` находится вне "всеобъемлющего" тега `<html>`.
Этот *мета-тег* задаёт, во-первых, версию языка HTML, на которой написана web-страница, а во-вторых, разновидность данной версии.
В случае HTML5 тег `<!doctype>` выглядит так:
```
<!doctype html>
```
Код любой web-страницы, написанной на HTML5, должен включать в самом начале этот мета-тег. В противном случае web-обозреватель будет показывать страницу в режиме совместимости и может не обработать правильно какие-либо её элементы.
HTML-документ состоит из двух разделов — заголовка (между тегами `<head>` и `</head>`) и содержательной части (между тегами `<body>` и `</body>`).
С учётом минимально необходимой дополнительной информации, такой как язык, кодировка и заголовок окна документа, правильный документ HTML5 будет выглядеть следующим образом (предполагается, что текст сохранён в кодировке UTF-8):

```
<!doctype html>
<html lang="ru">
 <head>
  <meta charset="utf-8">
  <title>Заголовок окна</title>
 </head>
 <body>
  Содержимое документа
 </body>
</html>
```

1.7. Раздел HEAD

Раздел `head` содержит техническую информацию о странице — заголовок, её описание и ключевые слова для поисковых машин, данные об авторе и времени создания страницы, базовом адресе страницы, кодировке и т.д.
Единственным обязательным элементом в разделе `head` является `<title>`. Текст, расположенный между тегами `<title>` и `</title>`, отображается в строке заголовка web-браузера. Длина заголовка должна быть не более 60 символов, иначе он полностью не поместится в заголовке браузера.
`<title>Заголовок страницы</title>`
Очень часто текст между тегами `<title>` и `</title>` используется в результатах, выдаваемых поисковым сервисом в качестве текста ссылки на эту страницу. По этой причине заголовок должен максимально полно описывать содержание страницы. Не следует писать что-то вроде "Главная страница", "Первая страница" и т.п.

Тег `<head>` может содержать также следующие элементы:

- `meta`: Одиночный тег, не требует парный закрывающий тег в конце. С помощью `<meta>` можно сообщать браузеру, поисковому роботу или другому устройству различную служебную информацию (или метаинформацию) о вашем сайте: кодировку текста, описание контента и так далее. Для этого используются теги `<meta>` с разными атрибутами и их значениями.
  Кодировка текста HTML-страницы указывается с помощью атрибута `charset`:
  `<meta charset="название кодировки">`
  Самая распространённая современная кодировка — `utf-8`. Это разновидность стандарта Unicode, наиболее часто используемая для web. Желательно включать этот тег всегда и располагать его первым в `meta`.
  Перечень ключевых слов задаётся тегом `<meta>`, у которого атрибут `name` имеет значение `keywords`. Ключевые слова (самые важные слова из содержания страницы) перечисляются в атрибуте `content` через запятую:
  `<meta name="keywords" content="важные, ключевые, слова">`
  Краткое описание (или аннотация) страницы задаётся похожим образом, только значение атрибута `name` меняется на `description`:
  `<meta name="description" content="краткое описание">`

- `link`: для подключения стилей к странице существует тег <link>. Для этого у него есть атрибут `href` в котором задаётся адрес стилевого файла, а значение `stylesheet` атрибута `rel` говорит браузеру, что мы подключаем именно стили, а не что-то другое.
```
<head>
  <link href="адрес_файла_стилей.css" rel="stylesheet">
</head>
```
- `style`: позволяет подключить стили для текущего документа
- `script` :позволяет подключить скрипты (прежде всего, программы на JavaScript), выполняемые в текущем документе.

1.8. Раздел BODY

Тег `body` ограничивает тело документа. Основная структура документа HTML всегда состоит из заголовка и тела. Формально нет необходимости явно помещать тело в элемент `body`, однако делая так, можно получить валидный документ и специфицировать атрибуты, влияющие на представление документа в целом (например, установить фоновое изображение или цвет).

1.9. Разметка средствами HTML5

Внутри <body> находятся те теги, которые отображаются на странице.

Тег **<main>** выделяет основное содержание страницы, которое не повторяется на других страницах.
Спецификация не допускает использование на одной странице более одного тега <main>, если у них нет специального атрибута `hidden`. Этот атрибут добавляется HTML-элементу, например, в одностраничных приложениях (Single Page Application), чтобы менять содержимое страницы, делая видимым тот или иной <main> в разных состояниях приложения. Атрибут `hidden` указывает браузеру, что элемент не должен отображаться и использоваться в момент, когда отображается и используется содержимое другого <main>.

Тег **<header>** содержит вводную часть страницы, которую чаще называют "шапкой", а тег **<footer>** описывает заключительную часть страницы, или "подвал". Существует тег **<section>**, который обозначает крупный смысловой (или "логический") раздел.

Тег **<article>** обозначает цельный, законченный и самостоятельный фрагмент информации.

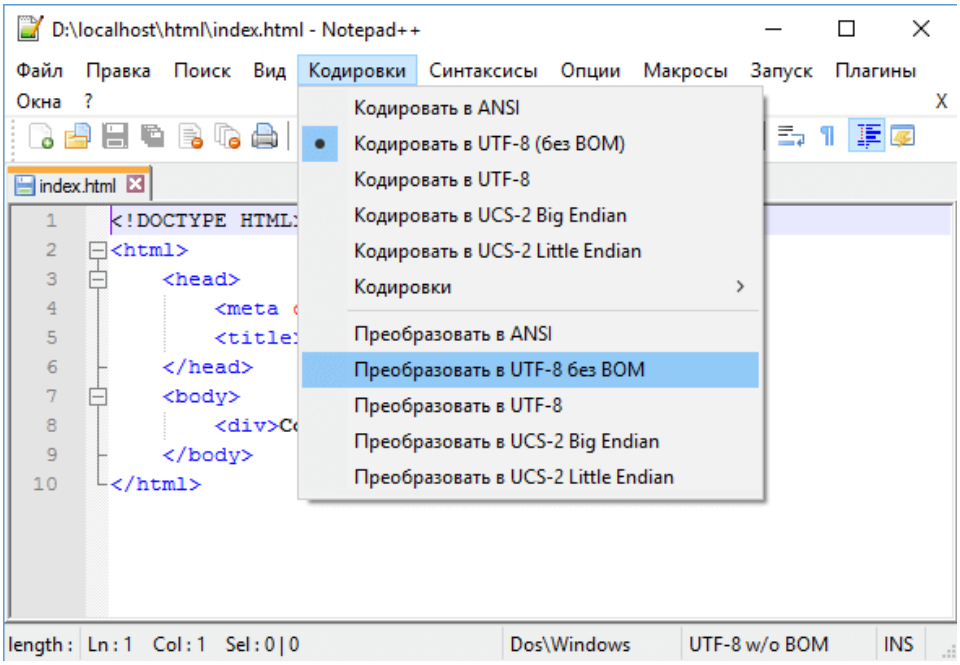
Для создания логического раздела с основной навигацией предназначен тег **<nav>** (сокращение от английского "navigation"). Обычно в <nav> включают ссылки на другие страницы или навигацию по текущей странице.

Тег **<aside>** включает в себя дополнительное содержание, не связанное напрямую с основным. Такие блоки ещё часто называют "сайдбарами" или боковыми панелями.

Пример простого макета HTML5:

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="utf-8">
    <title>Сайт начинающего верстальщика</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <header>
      <h1>Сайт начинающего верстальщика</h1>
    </header>
    <main>
      <nav>
        Навигация
      </nav>
      <section>
<p>Всем привет! Добро пожаловать на мой первый сайт.</p>
<p>Моё первое задание — вести дневник и писать обо всех своих
свершениях.</p>
      </section>
      <section>
        Раздел про навыки
      </section>
    </main>
    <footer>
      Подвал сайта
    </footer>
  </body>
</html>
```

Помните, что указанная в элементе `meta` кодировка должна совпадать с кодировкой самого документа. Как правило, текстовый редактор позволяет указать кодировку документа. Если мы хотим ориентироваться на `utf-8`, то в настройках текстового редактора надо выбирать "UTF-8 без BOM". Например, выбор кодировки в Notepad++ выглядит следующим образом:



1.10. Физическая и логическая разметка документа средствами, совместимыми с HTML4

В HTML4 различали *логическое* и *физическое* форматирование текста. Ниже приводятся основные теги каждой группы. Все эти теги должны закрываться и действуют на текст между открывающей и закрывающей частями тега.

Основные теги логического форматирования.

- `<em>` - выделение важных фрагментов курсивом
- `<strong>` - выделение важных фрагментов полужирным
- `<ins>` - выделение фрагмента подчеркиванием, когда требуется показать явно, что текст был вставлен после опубликования документа.
- `<del>` - выделение фрагмента перечеркиванием, когда требуется показать явно, что текст был удален после опубликования документа.
- `<cite>` - выделение цитат курсивом
- `<code>` - отображение фрагментов программного кода моноширинным шрифтом
- `<kbd>` - текст, вводимый с клавиатуры: отображается моноширинным шрифтом
- `<var>` - название переменных: отображается курсивом
- `<samp>` - выделение нескольких символов моноширииным шрифтом
- `<dfn>` - определение вложенного термина курсивом
- `<abbr title="Какое-то слово">` - аббревиатура (сокращение)
- `<acronym title="Какое-то слово">` - акроним (устоявшееся сокращение, используемое как отдельное слово)
- `<q lang="ru">` - определение кавычек

Основные теги физического форматирования.

- <i> - курсив
- <b> - полужирный
- <u> - подчеркнутый
- <strike> или <s> - перечеркнутый
- <tt> - моноширинный шрифт
- <big> - увеличить шрифт
- <small> - уменьшить шрифт
- <sup> - нижний индекс
- <sub> - верхний индекс

1.11. Абзацы и заголовки

Абзац формируется при помощи тега … . Атрибут align тега задает способ выравнивания абзаца (left, right, center, justify). Например, если необходимо сделать абзац с выравниванием по ширине, то надо написать:

```
<p align="justify">Содержание абзаца </p>
```

Заголовки позволяют, во-первых, озаглавить раздел, к которому относятся, а во-вторых, с помощью заголовков легко регулировать размер текста. Чем выше уровень заголовка, тем больше размер шрифта. Самым верхним уровнем является уровень 1 (<h1>), а самым нижним — уровень 6 (<h6>).

Также следует отметить, что использование заголовков приветствуется поисковыми системами.

Пример использования заголовков:

```
<h1>Заголовок первого уровня</h1>
<h2>Заголовок второго уровня</h2>
```

1.12. Разметка фрагментов текста

Символы-мнемоники

Это особые строки, которые начинаются с амперсанда (&) и заканчиваются точкой с запятой (;). Например, знак *меньше* на страницу можно вставить мнемоникой < (less than), а знак *больше* мнемоникой > (greater than):

Некоторые символы в HTML зарезервированы, то есть браузер считает их HTML-кодом. Например, любой текст после знака *меньше* (<) браузер будет пытаться интерпретировать как тег и на странице не отобразит. Чтобы использовать специальные символы в тексте страницы как обычные символы их нужно заменить на символы-мнемоники.

```
&lt;ul&gt;
&lt;/ul&gt;
```

В ряде случаев также двойная кавычка заменяется на `&quot;` , а амперсанд заменяется на `&` Существует множество других символов-мнемоник. Например: `&copy;`, `&laquo;`, `&raquo;`, `&sect;`. Полный список мнемоник представлен на специальной странице. А это – только некоторые:

		Неразбиваемый пробел	
"	"	Прямая кавычка	"
&	&	Амперсанд	&
<	<	Знак "меньше"	<
>	>	Знак "больше"	>
©	©	Копирайт	©
®	®	Зарегистрировано	®
™	™	Торговая марка	™

Перенос строк

Тег `<br>` (сокращение от "line break"). Применяется, чтобы вставить в текст перенос строки, не создавая при этом абзац. Например, при разметке стихов или текстов песен.

Тег `<nobr>` уведомляет браузер отображать текст без переносов. Если этого тега в коде документа нет, а также имеются переводы строки, они игнорируются и текст выравнивается по левому краю окна браузера или родительского элемента. При этом браузер переводы строк расставляет автоматически, чтобы текст полностью поместился по ширине окна. Использование тега `<nobr>` заставляет отображать текст без переносов, одной строкой, что может привести к появлению горизонтальной полосы прокрутки.

Верхний и нижний индексы

Теги `<sup>` и `<sub>`. Названия образованы от слов "superscript" и "subscript".

Тег `<sup>` отображает текст в виде верхнего индекса, а `<sub>` отображает текст в виде нижнего индекса.

Их используют для указания единиц измерения или для написания простых формул:

```
20м<sup>2</sup>
H<sub>2</sub>O
X<sup>3</sup>+X<sup>2</sup>=1
```

Для создания более сложных формул, эти теги можно использовать внутри друг друга.

Дата и время

Тег `<time>`. С помощью него можно описывать даты одновременно и для человека, и для машины. Для указания даты в машиночитаемом формате ISO 8601 существует атрибут `datetime` и выглядит так:

```
<time datetime="2016-11-18T09:54">09:54 утра</time>
<time datetime="2015-11-18">18 ноября 2015</time>
<time datetime="2018-09-23">в прошлую субботу</time>
<time datetime="2017-04-20">вчера</time>
```

Браузер отображает только содержимое тега, а содержимое `datetime` не отображается.

Акцентирование внимания

Теги и <i>. Названия образованы от слов "emphasis" и "italic". Предназначены для акцентирования внимания на слово или фразу. Визуально оба тега одинаковы, они выделяют текст курсивом.

Тег определяет текст, на который сделан *особый акцент*, меняющий смысл предложения.

```
Я <em>просто обожаю</em> холодные зимние дни!
```

Тег <i> применяется для обозначения текста, который отличается от окружающего текста, но не является более важным. Например, в <i> можно заключать *названия*, *термины*, *иностранные слова*. Также в этот тег можно обернуть *мысли* героя. В речи такой текст обычно выделяется интонационно:

```
Он взглянул в окно и подумал — <i>такого просто не может
быть</i>!
```

Выделение и придание важности

Теги и <b>. Название <b> образовано от слова "bold". Отображаются оба тега одинаково, они выделяют текст жирным.

Тег указывает на **важность** отмеченного текста. Он может использоваться для выделения предупреждений или части документа, которую пользователь должен увидеть раньше остального. При этом обозначение части текста тегом не должно изменять смысла предложения.

```
<strong>Внимание!</strong> Это место опасно. <strong>Вы можете
упасть в пропасть</strong>, если подойдёте близко к краю.
```

Тег <b> предназначен для выделения текста с целью привлечения к нему внимания, но без придания ему особой важности. Использовать его нужно только в случае, когда остальные теги выделения не подходят. Типичный пример — выделение вводного предложения статьи.

```
Вы входите в небольшую комнату. Ваш <b>меч</b> загорается
ярче. <b>Крыса</b> стремительно пробегает вдоль стены.
```

Описание изменений

Теги <del> и <ins>. Названия тегов образованы от слов "delete" и "insert". Предназначены для описания изменений в документе.

Тег <del> выделяет текст, который был удалён в новой версии документа. В браузере этот текст перечёркивается.

Тег <ins> выделяет текст, который был добавлен в новой версии документа. В браузере этот текст подчёркивается.

```
<ul>
  <li>Почистить посудомоечную машину</li>
  <li><del datetime="2009-10-11T01:25-
07:00">Погулять</del></li>
  <li><del datetime="2009-10-10T23:38-
07:00">Поспать</del></li>
  <li><ins>Купить принтер</ins></li>
</ul>
```

Разделение контента

Теги и . Это "чистые" элементы, и обычно они отлично подходят в качестве обёртки для стилизации или группировки других элементов. Использовать эти теги рекомендуется в тех случаях, если более подходящих семантических тегов не нашлось.

Тег (*раздел*) используется для группировки структурных элементов или в качестве вспомогательных контейнеров для создания нужной раскладки.

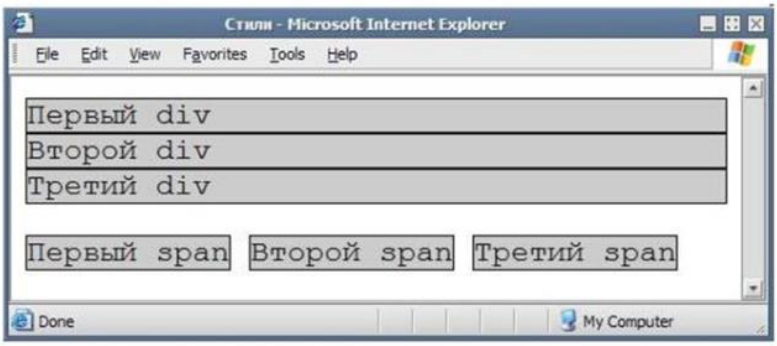
Тег (*фрагмент*) используется для группировки текстовых элементов, выделения отдельных слов или фраз внутри абзацев, пунктов списка и так далее.

```
<article>
  <div class="highlight">
    <p>…</p>
    <p>…</p>
  </div>
  <p>Текст, в котором <span>выделена фраза</span></p>
</article>
```

В показанном ниже примере отображение разделов и фрагментов управляется стилевыми классами, описание которых в листинге не приведено:

```
<div id="main">
      <div class="one">Первый div</div>
      <div class="one">Второй div</div>
      <div class="one">Третий div</div>

      <span class="two">Первый span</span>
      <span class="two">Второй span</span>
      <span class="two">Третий span</span>
</div>
```



Преформатированный текст и код

Тег <pre> (сокращение от "preformatted text"). Используется для отображения примеров кода, также применяется для отображения картинок ASCII Art. Браузер сохраняет и отображает все пробелы и переносы, которые есть внутри тега <pre>.

```
<pre>Пример
        преформатированного
        текста     с сохранёнными пробелами
                     и переносами строк</pre>
```

Тег <code> используется для обозначения фрагментов кода.

С его помощью размечается любой фрагмент текста, который распознается компьютером: код программы, разметки, название файла и так далее. Обычно браузеры отображают текст в теге <code> моноширинным шрифтом.

```
Тег <code>ul</code> — это неупорядоченный список.
```

Тег <code> можно вкладывать внутрь тега <pre>. Главное — соблюдать правильный порядок вложенности. Тег <code> можно вкладывать внутрь тега <pre>, а наоборот делать нельзя. Пример:

```
<pre><code>
if (a &gt; b) {                // Пример кода
  console.log('Hello!');  // на языке Яваскрипт
}
</code></pre>
```

Никакой разницы в отображении просто тега <pre> и тега <pre> с вложенным <code> в браузере не будет, но второй вариант разметки информативнее и выразительнее.

1.13. Цитирование

Небольшие цитаты

Тег <q> (сокращение от "quote"). Предназначен для выделения цитат внутри предложения. Текст внутри тега браузер автоматически обрамляет кавычками, поэтому добавлять кавычки вручную не нужно.

Источник цитат

Тег <cite>. В нём можно указывать помимо адреса источника цитаты ещё и название произведения, откуда цитируется текст, а также имя автора или организации, чей текст цитируется. Содержимое <cite> в браузере выделяется курсивом.

```
<p>По словам <cite>Чарльза Буковски</cite> — <q>Интеллектуал о простой вещи говорит сложно — художник сложную вещь описывает простыми словами.</q></p>
```

Тег <cite> может быть самостоятельным и не привязываться к цитате:

```
<p>Какой доктор ваш любимый (в сериале <cite>Доктор Кто</cite>)?</p>
```

Длинные цитаты

Тег <blockquote>. Предназначен для выделения длинных цитат, которые могут состоять из нескольких абзацев. Тег выделяет цитату не как фрагмент текста в предложении, а как отдельный блок текста с отступами.

```
<blockquote>
  <p>Ум ценится дорого, когда дешевеет сила.</p>
  <cite>Джейсон Стэтхэм</cite>
</blockquote>
```

В браузере контенту тега <blockquote> обычно добавляется дополнительный отступ слева и справа.

1.14. Тег комментария <!-- -->

Добавляет комментарий в код документа. Текст комментария не отображается на странице. Разрешается внутрь комментария добавлять другие теги, но вложенные комментарии (когда один комментарий расположен внутри другого) недопустимы.

Проверка валидности разметки HTML5: https://validator.w3.org
Проверка браузера на степень поддержки HTML5: http://html5test.com/

Задание к лабораторной работе 1:

Сделайте валидный документ HTML5, содержащий:

- корректную структуру документа;
- мета-теги с описанием документа;
- заголовки;
- теги логического и физического форматирования, включая смысловые и шрифтовые выделения, цитаты, исправления, листинги и т.д.

Содержимое документа – любое, например, краткий рассказ о себе или каком-либо предмете (явлении).
Проверьте работу документа в основных браузерах и его валидность.
Есть ли отличия в отображении элементов? С чем они, по-Вашему, связаны?
Сохраните пустой валидный файл HTML5 в качестве шаблона для последующей работы.