HTML (Hypertext Markup Language)

– это язык разметки, используемый для включения текстовых документов в Web-страницы.

Язык HTML в 1991 году создал Тим Бернерс-Ли (Tim Berners-Lee) в качестве простого способа передачи смысла и структуры гипертекстовых документов.

XHTML - это переработка HTML в соответствии с требованиями XML. Иными словами, в нем используется тот же словарь (те же элементы и атрибуты), как в HTML, но синтаксические правила взяты из XML, который является более строгим языком, чем HTML.

W3C

Видя необходимость упорядочить разработку HTML, Бернерс-Ли в 1994 году основал World Wide Web Consortium (W3C).

W3C продолжает надзирать за HTML и связанными с ним Web-технологиями и выпускает обновленные и стандартизованные версии HTML в виде публикаций, которые с 1995 года называются рекомендациями (Recommendations).

Роль HTML

Говорят, что размеченный HTML-документ образует ***структурный уровень*** Web-страницы.

Это основа, над которой надстраиваются ***уровень представлений*** (инструкции по передаче и отображению элементов) и ***уровень поведения*** (скрипты и интерактивная работа).

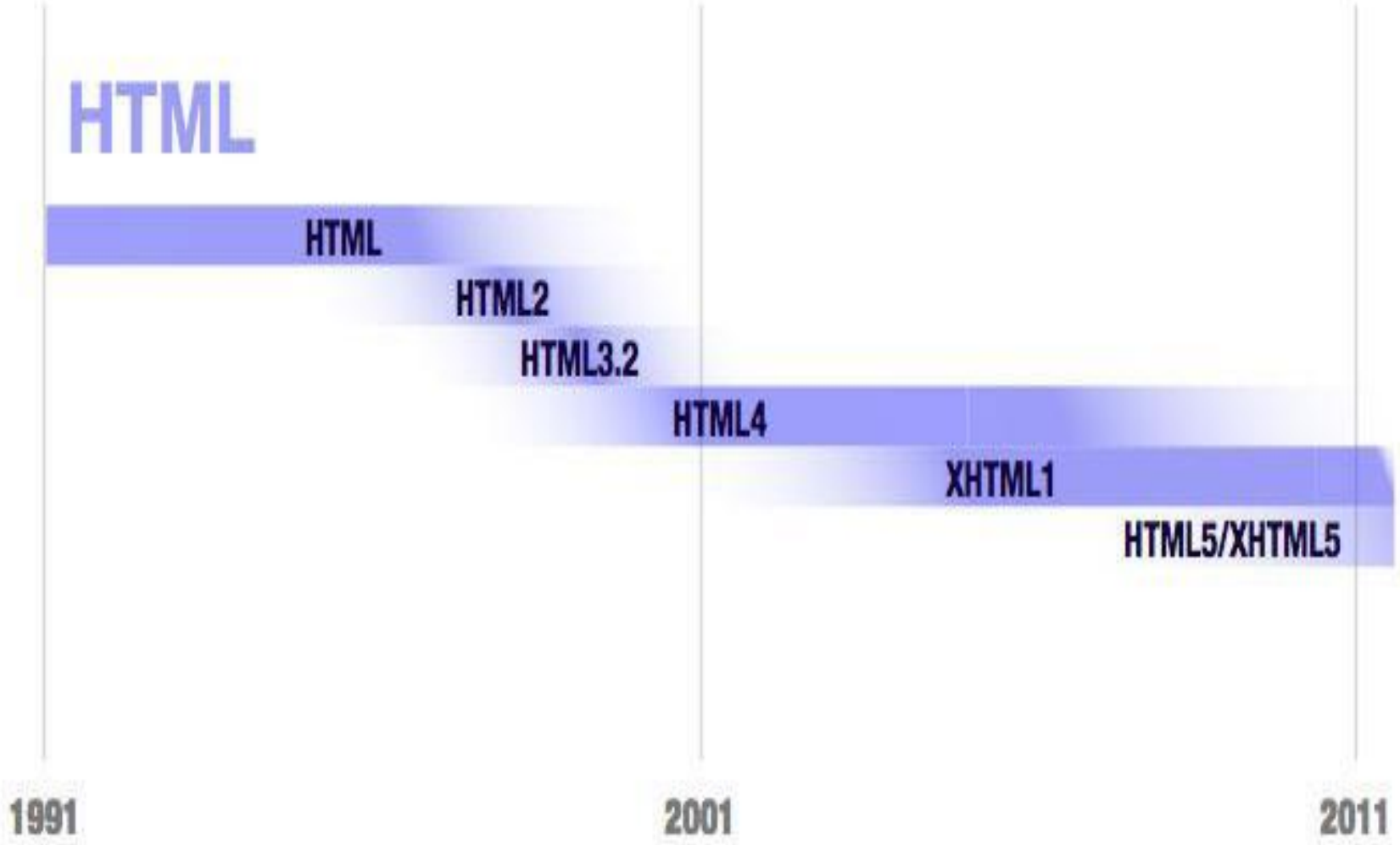
Отделение представления от структуры документа

До HTML существовал язык SGML **(Standard Generalized Markup Language),** который представлял собой сложный язык для описания структуры документов независимо от их внешнего вида. SGML - это обширный набор правил разработки языков разметки, таких, как HTML, но он столь всеобъемлющ, что HTML использует лишь малую часть его возможностей.

История HTML

- Предполагалось, что язык HTML уйдет в небытие, не дожив до XXI столетия. Организация ***W3C (World Wide Web Consortium - Консорциум Всемирной паутины)***, которая занимается разработкой и внедрением официальных стандартов Всемирной паутины, забросила язык HTML в далеком 1998 г., считая его не способным на дальнейшее выживание.
- Свои надежды на будущее консорциум W3C возлагал на модернизированного наследника HTML — язык XHTML.

- Но язык HTML не умер.
- В 1998 г. консорциум W3C прекратил его поддержку и попытался заменить его языком на основе языка XML - XHTML 1.0.


HTML
HTML
HTML2
HTML3.2
HTML4
XHTML1
HTML5/XHTML5
1991
2001
2011

- Компании Opera, Mozilla и Apple создали группу ***WHATWG (Web Hypertext Appliccation Technology Working Group - рабочая группа по технологии гипертекстовых веб-приложений)*** с целью работы над новыми решениями.
- Группа не ставила перед собой задачу заменить HTML, ее целью было плавное расширение языка и обратная совместимость. Надо сказать, что самая ранняя версия работы этой группы включала две спецификации расширений — *Web Applications 1.0* и *Web Forms 2.0*. В конечном итоге эти стандарты эволюционировали в HTML5.

html 5

- HTML 5 – это, скорее, новая платформа для создания веб–приложений, нежели стандарт продолжающий традиции предшественников.
- HTML 5 регламентирует взаимодействие с JavaScript посредством объектной модели документа.
- HTML 5 поддерживает все элементы HTML 4.

Что входит в состав HTML5

- ***Ядро HTML5***
- Эта часть HTML5 составляет официальную версию спецификации организации W3C. Она содержит новые семантические элементы, новые и усовершенствованные элементы управления для веб-форм, поддержку аудио и видео, а также холст для рисования с помощью JavaScript.

Ранние возможности HTML5

- Это возможности, которые были реализованы в первоначальной спецификации HTML5, подготовленной группой WHATWG. Большинство из них — это спецификации для возможностей, требующих JavaScript и поддерживающих развитые веб-приложения. Наиболее важными являются локальное хранение данных, приложения, работающие в автономном режиме, и обмен сообщениями.

Возможности, иногда называемые HTML5

- Это возможности следующего поколения, которые часто считаются частью HTML5, хотя они никогда не входили в стандарт HTML5. Эта категория включает спецификацию CSS3 и геолокацию.

группа WHATWG продолжает свою работу, придумывая будущие возможности HTML. Только теперь она называет его не HTML5, а просто HTML, объясняя это тем, что HTML будет продолжать существовать, как живой язык.

Особенности проектирования современных веб – решений

- **Пользователь нетерпелив**, не мотивирован на чтение больших блоков информации и статей. Необходимо следование методикам захвата и удержания внимания.
- **Дружественный интерфейс**. Пользователь разборчив, привык к простоте и быстрым действиям. То есть, у пользователя не должно возникать вопросов, как осуществить то, или иное действие (ответить на сообщение, оценить товар и т.д.).
- **Однородность**. Большинство решений (социальные сети, форумы, новостные ленты и т.п.) уже имеет устоявшиеся форматы и формы представления. Не стоит пытаться внести что – то оригинальное непосредственно в основной функционал сайта, велик риск того, что пользователь предпочтет знакомое новому.

Кроме того, следует помнить, что:

- Сайт должен загружаться быстро.
- Сайт должен быть оптимизирован для выполнения основной своей задачи.
- Содержание сайта должно соответствовать его оформлению.
- Информация должна быть легко читаема.
- Не должно быть элементов, затрудняющих восприятие информации.
- На сайте не должно быть грамматических ошибок.

Элементарная структура документа

- В данном примере показана элементарная структура XHTML-документа, как она определяется в рекомендации XHTML 1.0.

- **<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">**
- **<html xmlns="http://www.w3.org/1999/xhtml">**
- И в HTML5:

- **<!DOCTYPE html>**
- **<html>**
- **<head>**
- **<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />**
- **<title> Название документа</title>**
- **</head>**

- **<body>**
- **Содержимое документа**
- **</body>**
- **</html>**

Требования XHTML

- XHTML (будучи XML-приложением) требовал строгих правил синтаксиса XML-разметки. XHTML документы отличаются от HTML-документов тем, что в первом случае вы должны быть абсолютно уверены, что документ выполняет синтаксические правила XML (иными словами, является *хорошо сформированным).*

Перечислим атрибуты, которые являлись сокращенными в HTML, но для которых в XHTML требуются значения

- **checked=«checked»**
- **disabled=«disabled»**
- **noresize=«noresize»**
- **compact=«compact»**
- **ismap=«ismap»**
- **nowrap=«nowrap»**
- **declare=«declare»**
- **multiple=«multiple»**
- **readonly=«readonly»**
- **defer=«defer»**
- **noshade=«noshade»** **selected=«selected»**

Пустые элементы

<area />
<base />
<basefont />

<col />
<frame />
<hr />
<img />
<input />
<isindex />
<link />
<meta />
<param />

Заголовок документа

Каждый элемент **head** должен включать в себя элемент **title**, который содержит описание документа. Элемент **head** также может включать любой из следующих элементов в любом порядке: **script, style, meta, link, object, isindex** и **base**. Элемент head служит просто контейнером для этих элементов и не имеет своего собственного содержимого.

Другие элементы заголовка

base

Этот элемент определяет базовое местоположение документа, которое служит ссылкой для всех путей и ссылок документа.

isindex

Устарел. Этот элемент когда-то использовался для добавления на страницу простой поисковой функции. Он был объявлен устаревшим в HTML 4.01 с заменой на элементы ввода для форм.

link

Этот элемент определяет взаимосвязь между текущим документом и другим документом.

script

С помощью этого элемента в заголовок документа можно добавить код JavaScript или VBScript.

style

Еще одним методом связывания таблицы стилей с HTML-документом является встраивание его в заголовок документа при помощи элемента style.

meta

Элемент meta используется для указания информации о документе, например такой, как ключевые слова или описания, которые помогают поисковым системам.

<meta />

Атрибуты

- id="text" (только XHTML)
- content="text"
- http-equiv="text"
- name="text"

Данные, включаемые в элемент meta, могут использовать серверы, Web-браузеры и поисковые системы, но для читателя они невидимы. Этот элемент всегда должен помещаться в элемент head документа.

- В документе может использоваться несколько элементов meta. Существует два типа элементов meta, в которых используется либо элемент name, либо элемент http-equiv.
- В любом случае необходим атрибут content, в котором указывается значение (или значения) именованной информации или функции. В приведенном примере показан синтаксис обоих типов элементов meta:
- <meta http-equiv="name" content="content" />
- <meta name="name" content="content" />

http-equiv

- Информация, указанная в атрибуте http-equiv, обрабатывается так, как если бы она пришла в заголовке HTTP-ответа. Заголовки HTTP сдержат информацию, которую сервер передает браузеру непосредственно перед отправкой HTML-документа, например информацию о типе данных (media type), и другие значения, влияющие на работу браузера. Следовательно, атрибут **http-equiv** предлагает информацию, которая будет, в зависимости от описания, изменять способ обработки документа браузером.

name

Атрибут name используется для вставки скрытой информации о документе, которая не соответствует HTTP-заголовкам. Например:

- <meta name="author" content="Jennifer Niederst Robbins" />
- <meta name="copyright" content="2006/ O'Reilly Media" />

Указание типа информации и кодировки символов

- Рекомендуется (хотя и не является обязательным), чтобы тип информации и кодировка символов указывались внутри документов (X)HTML, чтобы эта информация сохранялась в документе

```
<meta   http-equiv="content-type"
  content="text/html;
 charset=UTF-8"   />
```

Другие области использования http-equiv

expires

Указывается дата и время, по истечении которых документ будет считаться устаревшим. Web-роботы могут использовать эту информацию для удаления устаревших документов из индексов поисковых систем. Используется формат времени и даты, соответствующий стандартному формату для заголовков HTTP, поскольку предполагается, что атрибут http-equiv повторяет поле заголовка HTTP.

<meta http-equiv="expires" content="Wed 12 Jun 2001 10:52:00 EST" />

content-language

Это значение может использоваться для указания языка, на котором написан документ. Браузер может послать соответствующий заголовок Accept-Language, который заставит сервер выбрать документ, написанный на соответствующем языке.

В данном примере для браузера указывается, что естественным языком документа является французский:

```
<meta   http-equiv="content-language"
   content="fr"   />
```

Имена в элементе meta для поисковых систем

<meta name="description" content="Toropova Olga' resume and web design samples" />

<meta name="keywords" content="designer, web design, branding, logo design" />

<meta name="author" content=" Toropova Olga" />

<meta name="copyright" content="2009, O'Reilly Media" />

Тело документа

<body>...</body>

Атрибуты

Базовые атрибуты: id, class, style, title

Внутренние события: onload, onunload, onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkey-down, onkeyup

Устаревшие атрибуты

alink="#rrggbb" или "имя цвета"

background="URL"

bgcolor="#rrggbb" или "имя цвета"

link="#rrggbb" или "имя цвета"

text="#rrggbb" или "имя цвета"

vlink="#rrggbb" или "имя цвета"

Текстовые элементы делятся на две главные категории: **внутристрочные (inline) и блочные.**

Внутристрочных элементы встречаются в потоке текста и по умолчанию не приводят к переводу строки.

Блочные элементы по умолчанию начинаются в представлении с новой строки и состыковываются в обычном потоке документа как блоки.

Блоковые элементы

h1-h6	Заголовок
p	Абзац
pre	Предварительно форматированный текст
address	Контактная информация
blockquote	Длинная цитата

Внутристрочные элементы

- **em** – обозначает выделенный текст. Элементы **em** почти всегда отображаются курсивом.
- **strong** – обозначает сильно выделенный текст. Элементы **strong** почти всегда отображаются полужирным текстом.
- **abbr** – обозначает сокращенную форму.
- **cite** – обозначает цитату - ссылку на другой документ, в особенности на книги, журналы, статьи и т. п. Эти элементы обычно отображаются курсивом.
- **q** – **о**бозначает краткую внутристрочную цитату.
- **dfn** – обозначает определение или первое вхождение заключенного в тег термина. Может использоваться для привлечения внимания к появлению специальных терминов и фраз. Определения терминов часто отображаются курсивом.
- **code** – обозначает фрагмент программного кода. По умолчанию код отображается в браузере специальным шрифтом фиксированной ширины (обычно - Courier).
- **kbd** – является сокращением от слова «keyboard» (клавиатура) и обозначает текст, вве денный пользователем. Может использоваться в технических документах. Как правило, такой текст отображается шрифтом фиксированной ширины.
- **samp** – обозначает пример выходных данных программы, скрипта и т. п. Может использоваться в технических документах. Как правило, такой текст по умолчанию отображается шрифтом фиксированной ширины.
- **var** – обозначает экземпляр переменной или аргумент программы. Это еще один элемент, который наиболее полезен в технической документации. Переменные обычно отображаются курсивом.
- **sub -** Подстрочный
- **sup -** Надстрочный
- Элементы следует выбирать по смыслу, а не по визуальному эффекту в браузере. Элементы фраз могут содержать другие внутристрочные элементы.

br **Перенос строки**

Обозначение редактирования

ins Вставленный текст
del Удаленный текст

Общие элементы

Общие элементы **div** и **spa**n предоставляют авторам возможность создавать собственные элементы. Элемент **div** применяется для обозначения блочных элементов, а элемент **span** - для обозначения внутристрочных элементов.

div Блок
span Участок внутристрочного текста

Списки

ul Несортированный список
ol Сортированный список
li Элемент списка
dl Список определений
dt Термин
dd Определение
menu Список меню
dir Директория

Стилевые элементы

b Полужирный
big Текст большего размера
small Текст меньшего размера
i Курсив
s Зачеркнутый
strike Зачеркнутый
tt Телетайп
u Подчеркнутый
font Гарнитура шрифта, цвет и размер
basefont Задает размер шрифта по умолчанию
nobr Нет переноса строки
wbr Перенос слова
hr Горизонтальная линейка

Списки

- Несортированная информация (маркированный).
- Сортированная информация (нумерованный).
- Термины и определения

Несортированные списки

Несортированные списки, а также их пункты представляют собой блочные элементы, так что каждый из них отображается начиная с новой строки.

Устаревшие атрибуты

type="**disc** | **circle** | **square**“

Синтаксис несортированных списков

Пример

```
 <ul type=“square” >
     <li>Unordered   information</li>
     <li>Ordered   information</li>
     <li>Terms   and   definitions</li>
</ul>
```

Сортированные списки

Устаревшие атрибуты

start="number"

type= “1| I | A | a | i "

Сортированные списки имеют ту же базовую структуру, что и несортированные, и это показано в следующем простом примере.

```
<ol type=“A”>
<li>Встать с кровати</li>
<li>Принять душ</li>
<li>Погулять с собакой</li>
</ol>
```

Списки определений

- Используйте *список определений* для тех списков, которые состоят из пар термин - определение.
- Элемент **dt** (термин) может содержать только внутристрочные материалы, но элемент **dd** может включать блочные или внутристроч-ные элементы.
- Все три элемента, используемые в списках определений, представляют собой блочные элементы, и они по умолчанию начинаются с новой строки.

Пример:

```
<dl>
<dt> Файл </dt>   <dd>Именованная область памяти</dd>
<dt>  Hardware </dt>   <dd>Технические средства</dd>
<dt> Software </dt>   <dd>Программные средства </dd>
</dl>
```

. <dl>
<dt>**em**</dt>
<dd>обозначает выделенный текст. Элементы em почти всегда отображаются курсивом.
</dd>
<dt>strong</dt>
<dd> обозначает сильно выделенный текст. Элементы Strong почти всегда отображаются жирным шрифтом.
</dd>
<dt>abbr</dt>
<dd> обозначает сокращенную форму.</dd>
<dt>acronym</dt>
<dd> обозначает аббревиатуру.</dd>
</dl>

iframe

Этот элемент в отличае от тега <frame /> остался в новой спецификации.

iframe – это отдельная интернет-страница, которая включается посредством тегов <iframe> </iframe> в другую страницу.

iframe может быть внедрен в любое место страницы. Его местоположение более точно регулируется с помощью Каскадных таблиц стилей.

iframe может внедряться без полос прокрутки и не иметь рамки.

Пример **HTML iframe**'а:

```
< head>
 < title>HTML iframe</title>
 < /head>
 <body>
 <iframe src="HTML.php" width="91%" height="323" frameborder="1"><
 /iframe>
 < /body>
 </html>
```

Атрибуты и значения

src="" – указывает url документа.

frameborder="" – граница вложенного документа. Значения: **1** – по умолчанию и **0** – границы нет.

scrolling="" – определяет наличие полосы прокрутки. yes – по умолчанию и no – полосы прокрутки нет.

width="" – определяет ширину **фрейма** в пикселях или в процентах.

height="" – определяет высоту фрейма в пикселях или в процентах.

align="" – определяет выравнивание. Возможные значения: top, right, bottom, left, middle**.**

Теги работы с текстом, появившиеся в HTML5

<section> </section>

Парный тег, задающий блок (секцию) элементов.

<header> </header>

Парный тег, определяющий начало документа (секции).

<footer> </footer>

Парный тег, определяющий окончание документа (секции).

Использование <header>

<header> <div class="header-bg"**>**
<img src="images/header-title.png" alt="Как поймать льва в пустыне" **/>**
** </header>**

Попытка добавить в стилях фон к тегу **<header>** ни к чему не привела, фон в некоторых браузерах отображаться не желает. Все новые теги следует сделать вначале блочными через свойство display, тогда они начнут корректно выводиться.

Пример <style> header {
display: block;
background: #00B0D8 **url(**images/header-gradient.png**)** repeat-x; }
</style>

Данный пример будет работать во всех браузерах, кроме IE7 и IE8. Internet Explorer не добавляет стиль к элементам, которые не понимает. Это недоразумение можно исправить, если создать фиктивный элемент с помощью JavaScript. Для этого включим в **<head>** такой код.

<script> document.createElement("header"); **</script>**

Если на странице встречается один тег, этот скрипт вполне подойдёт для работы. Но не хочется повторять строку десять раз для десяти разных тегов, поэтому автоматизируем этот процесс через цикл. Сами теги указываются списком.

Скрипт для IE

```
<!--[if lt IE 9]> <script>
var e = ("article, aside,figcaption,figure,footer,header, hgroup,nav, section, ime").split(','); for (var i = 0; i < e.length; i++)
        { document.createElement(e[i]); }
                </script>
<![endif]-->
```

Сам скрипт заключается в условные комментарии, чтобы выполнялся только для IE версии 8.0 и ниже. В IE9 поддержка новых тегов HTML5 уже включена.

Можно воспользоваться общедоступным скриптом написанным Реми Шарпом и распространяемым по лицензии MIT. Для этого достаточно указать на него ссылку, как показано в примере.

Скрипт для IE

```
<!--[if lt IE 9]>
<script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
<![endif]-->
```

Все указанные скрипты должны располагаться в коде перед CSS.

- **<main>…</main> (HTML5) – контейнер с основным контентом веб-**
- страницы.

<hgroup> </hgroup>

Парный тег, определяющий группу заголовков.

```
<html>    < body>
<hgroup>
   <h1>HTML справочник</h1>
   <h4>Новые теги</h4>                    </hgroup>
   <p>В спецификации HTML 5 представлены
   новые теги...</p>
   < /body>   < /html>
```

<time> </time>

Парный тег, определяющий время, или дату.

<nav> </nav> Парный тег, определяющий группу ссылок (навигационное меню).

<mark> </mark>

Парный тег, определяющий важную смысловую часть текста.

<aside></aside>

Парный тег, определяющий дополнительный контент. Служит для размещения ссылок, меток и т.п. на иную информацию, не относящуюся напрямую по смыслу к основному содержанию текста.

<article></article>

Парный тег. Тег представляет собой компонент страницы, содержащей контент, предназначенный для самостоятельного распространения. Иными словами в тег <article> могут быть помещены: форумный пост, статья, запись блога, комментарий или иной другой независимый элемент содержимого. Аналогично <section> используется для секционирования контента.

- <bdo> </bdo> Определяют направление текста
- Атрибуты **dir="rtl", dir="ltl"** – определяют направление текста. В первом случае направлевие будет справа налево, а во втором – слева направо (по умолчанию).
- title="" – всплывающая подсказка.

<command> </command>

отображают команды для выпадающих списков, кнопок:

- Теги <command> </command> должны быть размещены внутри <menu> </menu>.
- Теги <command> </command> впервые представлены в спецификации **HTML 5**.

<figure> … </figure>

Используется для группирования любых элементов, например, изображений и подписей к ним.

```
<!DOCTYPE html>   <html>   <head>
  <meta charset="utf-8" />         <title>Тег FIGURE</title>
  <script>
   document.createElement('figure');
   document.createElement('figcaption');
  </script>
  <style>
   figure {
    background: #5f6a72; /* Цвет фона */
    padding: 10px; /* Поля вокруг */
    display: block; /* Блочный элемент */
    width: 150px; /* Ширина */
    float: left; /* Блоки выстраиваются по горизонтали */
    margin: 0 10px 10px 0; /* Отступы */
    text-align: center; /* Выравнивание по центру */
   }
   figcaption {
    color: #fff; /* Цвет текста */
   }
  </style>
 </head>
 <body>
  <article>
   <figure>
    <p><img src="images/thumb1.jpg"
alt="" /></p>
    <figcaption>Софийский
собор</figcaption>
   </figure>
   <figure>
    <p><img src="images/thumb2.jpg"
alt="" /></p>
    <figcaption>Польский
костел</figcaption>
   </figure>
  </article>
 </body>
</html>
```

Теги <datalist> </datalist>

```
<!DOCTYPE HTML>
   < html>
   < body>
   < input list="computers" />
   < datalist id="computers">
   <option value="sony" />
   <option value="toshiba" />
   <option value="asus" />
   <option value="acer" />
   < /datalist>
   < /body>
   < /html>
```

- Теги <datalist> </datalist>, с обязательными в этом случае <option />, не видны в браузере, они лишь определяют допустимые значения при вводе данных посредством элемента <input />. Атрибут list="" тега <input /> указывает на функциональную связь между группой тегов.

Теги <details> </details>

Определяют детали документа

<!DOCTYPE HTML>

< html>

< body>

<details>Дата публикации: Февраль 2010</details>

< /body>

< /html>

Теги <dialog> </dialog>

```
<!DOCTYPE HTML>
    < html>
    < body>
    < dialog>
    < dt>Прохожий</dt>
    < dd>&mdash; Доброе утро!</dd>
    < dt>Пассажир</dt>
    < dd>&mdash; Доброе...</dd>
    < dt>Прохожий</dt>
    < dd>&mdash; Как ваши дела?</dd>
    < dt>Пассажир</dt>
    < dd>&mdash; Простите, а вы кто?</dd>
    < /dialog>
    < /body>
    < /html>
```

Требования по вложенности

- элемент a не может содержать другой элемент a;
- элемент pre не может содержать элементы img, object, applet, big, small, sub, sup, font или basefont;
- элемент form не может содержать другие элементы form;
- элемент button не может содержать элементы a, form, input, select, textarea, label, button, iframe или isindex;
- элемент label не может содержать другие элементы label.

HTML5. Работа с мультимедиа

- ** <img src="sample.jpg"> **

К числу необязательных атрибутов **<img>** относятся:

align – задает тип выравнивания изображения;

alt – задает текст, отображаемый в случае, если картинка не загрузилась;

border – определяет толщину рамки вокруг изображения;

height – задает высоту изображения;

hspace – задает величину горизонтального отступа от изображения до ближайшего контента;

ismap – определяет, является ли изображение картой (т.е. к различным частям изображения "привязаны" разные ссылки);

vspace – задает величину вертикального отступа от изображения до ближайшего контента;

width – задает ширину изображения;

usemap – определяет ссылку на <map>, содержащий координаты клиентской карты - изображения.

Основы работы с видео и звуком

В спецификации HTML5 предусмотрено два тега для работы с аудио и видео соответственно: <audio> и <video>.

Данные теги являются компонентами собственной среды браузера. Это означает, что не используется никаких сторонних средств для воспроизведения мультимедиа информации, что, во-первых, повышает безопасность, во-вторых, за счет более тесной интеграции, позволяет обходиться меньшим количеством аппаратных ресурсов для воспроизведения мультимедиа, и, в-третьих, позволяет избежать ряда проблем отображения информации (визуальное пересечение с остальным контентом).

Помимо этого, использование <audio> и <video> позволяет организовать управление из веб-сценариев.

Также существуют и недостатки указанных тегов, спецификация HTML5 поддерживает далеко не все кодеки, строго говоря, из спецификации исключены все упоминания об обязательной поддержке каких-либо кодеков.

Частичным решением проблемы кодеков может служить использование элемента <source>, позволяющий объявить несколько источников мультимедиа, из которых браузером выберет наиболее подходящий.

<video> </video>

```
<!DOCTYPE HTML>
    < html>
    < head>
    < meta http-equiv="Content-Type" content="text/html;
    charset=utf-8" />
    < /head>
    < body>
    < video src="video.mp4" controls="controls">
    Ваш браузер не поддерживает теги <video> </video>!
    Обновите версию браузера!
    < /video>
    < /body>
    < /html>
```

Тег <source /> определяет источник видео:

```
<video controls="controls">

<source src="video.m4v" type="video/mp4"
/> <!-- MPEG4 для браузеров Safari -->

< source src="video.ogg" type="video/ogg"
/>  <!-- Ogg Theora для Firefox -->

< /video>
```

Атрибуты и значения

autoplay="autoplay" – видео воспроизводится сразу после загрузки страницы.

autobuffer="autobuffer" – видео воспроизводится уже в момент загрузки страницы.

controls="controls" – панель управления видеоплеером.

loop="loop" – по окончанию, видео проигрывается снова.

src="url" – источник видео.

type="video/ogg" – определяет формат видео.

height="" – высота видеоплеера.

width="" – ширина видеоплеера.

poster ="" Указывает адрес картинки, которая будет отображаться, пока видео не доступно или не воспроизводится.

Атрибут preload

Используется для загрузки видео вместе с загрузкой веб-страницы. Этот атрибут игнорируется, если установлен **autoplay**.

Значения

none Не загружать видео.

metadata Загрузить только служебную информацию (размеры видео, первый кадр, продолжительность и др.).

auto Загрузить видео целиком при загрузке страницы.

Пример

```
<!DOCTYPE html>
<html> <head> <meta charset="utf-8">
    <title>preload</title> </head> <body>
    <video controls preload="metadata">
    <source src="video/duel.ogv" type='video/ogg;
    codecs="theora, vorbis"'>
    <source src="video/duel.mp4" type='video/mp4;
    codecs="avc1.42E01E, mp4a.40.2"'>
    <source src="video/duel.webm" type='video/webm;
    codecs="vp8, vorbis"'>
Тег video не поддерживается вашим браузером.
<a href="video/duel.mp4">Скачайте видео</a>. </video>
    </body> </html>
```

Теги <audio> </audio>

Текст, помещающийся внутри данного тега будет отображаться в браузерах, не поддерживающих <audio>.

```
<!DOCTYPE HTML>
    < html> < head>
    < title>Audio player в html: музыкальный поток на
    сайте</title>
    < /head>
    < body>
    < audio src="music.ogg" controls="controls">
    Ваш браузер не поддерживает теги <audio> </audio>!
    Обновите версию браузера!
    < /audio>
    < /body>  < /html>
```

Атрибуты и значения

autoplay="autoplay" – определяет воспроизведение музыкального файла сразу же после загрузки страницы.

autobuffer="autobuffer" – используется в паре с autoplay="autoplay" – определяет воспроизведение музыкального файла уже в момент загрузки страницы.

controls="controls" – показывает панель управления плеером.

src="url" – источник звукового файла.

- Строка: Ваш браузер не поддерживает теги <audio> </audio>! Обновите версию браузера! – будет показана если версия браузера пользователя не слишком новая.

Файл должен быть с расширением **.ogg**

- <audio autoplay controls src="1.mp3">
  Тег <audio> не поддерживается
  </audio>

Элемент **<embed>**

используется для загрузки и отображения объектов (например, видеофайлов, флэш-роликов, некоторых звуковых файлов и т.д.), которые исходно браузер не понимает.

<embed width="..." height="..."**></embed>**

Закрывающий тег не требуется.

Атрибуты

aalign Определяет как объект будет выравниваться на странице и способ его обтекания текстом.

hheight Высота объекта

hhidden Указывает, скрыть объект на странице или нет.

hhspace Горизонтальный отступ от объекта до окружающего контента.

ppluginspage Адрес страницы в Интернете, откуда можно скачать и установить плагин к браузеру.

ssrc Путь к файлу.

type MIME-тип объекта.

vspace Вертикальный отступ от объекта до окружающего контента.

wwidth Ширина объекта.

Пример

```
<!DOCTYPE html>
<html> <head> <meta charset="utf-8"> <title>Ter
    EMBED</title> </head> <body>
    <embed src="images/mouse.swf" width="
    400" height="300"  type="application/x-sho
    ckwave-flash"  pluginspage="http://www.m
    acromedia.com/go/getflashplayer">
</body> </html>
```