

Основы WEB-технологий

Введение в HTML



Лекция 1

Веб-страница — это текстовый документ, который обычно является представлением некоторого ресурса всемирной паутины (WWW — World-Wide Web).

Веб-страница может содержать списки, таблицы, графические изображения, а также ссылки на другие ресурсы. Благодаря наличию ссылок содержимое веб-страницы часто называют гипертекстом.

Подобно разработке программы создание веб-страницы состоит в написании исходного кода с той лишь разницей, что в первом случае исходный код записывается на некотором языке программирования, а во втором — на специальном языке разметки гипертекста (HTML — Hypertext Markup Language).

При написании исходного кода веб-страницы, так же, как и исходного кода программы, можно использовать как обычные текстовые редакторы типа Блокнота, так и специализированные редакторы и среды, поддерживающие подсветку синтаксиса, подсказки и автопродолжение.



1.1. Структура страницы

Приведем пример исходного кода простейшей веб-страницы.

```
<!DOCTYPE HTML PUBLIC
  "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Заголовок страницы</title>
  </head>
  <body>
    <p>Текст страницы</p>
  </body>
</html>
```

Исходный код страницы начинается с объявления типа, которое имеет вид `<!DOCTYPE ... >`. Объявление типа задает используемую версию языка HTML. Остальная часть исходного кода состоит из тегов и обычного текста. Теги могут быть **открывающими** и **закрывающими**. В данном примере открывающие теги — это `<html>`, `<head>`, `<title>`, `<body>` и `<p>`, а закрывающие — `</html>`, `</head>`, `</title>`, `</body>` и `</p>`.

Назначение элемента определяется его именем, поэтому имена элементов являются предопределенными и встроены в язык HTML подобно тому, как ключевые слова встроены в язык программирования. Также именем элемента определяется, какие другие элементы могут располагаться в данном элементе, а также может ли элемент содержать обычный текст.

Возвращаясь к вышеприведенному примеру, отметим, что:

- элемент `html` представляет всю страницу и должен содержать элементы `head` и `body`;
- элемент `head` служит для группировки различной служебной информации о странице, не составляющей самого ее содержания. элемент `head` должен включать элемент `title`;
- элемент `title` задает название страницы, отображаемое в заголовке окна браузера. Содержанием элемента `title` должен быть обычный текст, не содержащий элементов;
- элемент `body` задает само содержание страницы. Он должен содержать один или несколько блоковых элементов;
- элемент `p` является блоковым и задает абзац текста.

Для некоторых элементов закрывающий, а иногда и открывающий тег является необязательным и может быть пропущен в том случае, если это не вызывает неоднозначности. В частности, предыдущий пример может быть записан так:

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

В данном случае опущены оба тега элементов `html`, `head` и `body`, а также закрывающий тег элемента `p`. Несмотря на это, с точки зрения структуры, данная страница ничем не отличается от той, которая приведена ранее, т. е., например, в данной странице присутствует элемент `html`, содержащий элементы `head` и `body`.

1.2. Комментарии

В исходном коде страницы можно помещать комментарии.

Например:



```
<!-- этот текст является комментарием -->
```

Комментарии можно ставить в любом месте, в котором допустим пробел.

1.3. Сущности

Кроме тегов и обычного текста в исходном коде могут встречаться т. н. сущности. В отличие от тегов сущность обычно не определяет элемента, а служит для подстановки в текст символа, который либо имеет специальное значение (угловые скобки, амперсанд и двойные кавычки), либо отсутствует в используемом наборе символов (например, греческие буквы).

Приведем примеры:

Исходный код	Что получается
<code>1<2</code>	1<2
<code>&laquo;Знание &mdash; сила&raquo;</code>	«Знание — сила»

Исходный код	Что получается
<code>&alpha; &#946; &#x3B3;</code>	α β γ
<code>&#x65E5</code>	日

Сущности бывают именованными и числовыми. В первом случае символ задается именем, а во втором — кодом в соответствии с кодировкой Unicode.

В данном примере используются именованные сущности lt, laquo, nbsp, mdash, raquo и alpha, а также числовые сущности, соответствующие символам с кодами 946, $3B3_{16}$, $65E5_{16}$, 6708_{16} и $706B_{16}$.

При использовании числовой сущности код символа может задаваться как в десятичной, так и в шестнадцатеричной системе счисления (в последнем случае после знака решетки ставится буква x).

1.4. Строчные элементы

К строчным элементам относятся:

- `big`, `small` – увеличивают и уменьшают размер шрифта;
- `sup`, `sub` – преобразуют текст в верхний или нижний индекс;
- `em`, `strong` – выделяет текст, причем `STRONG` означает более сильное выделение, чем `EM`;
- `dfn` – текст определения чего-либо;
- `q` – текст цитаты;
- `cite` – источник цитаты.

Все перечисленные элементы могут содержать как обычный текст, так и другие строчные элементы.

Приведем пример:

Итак, `<dfn>`тип данных ` ``—` это ``множество`` значений`…``</dfn>`.

Также к строчным элементам относится элемент `br`, использующийся для переноса на новую строку. Этот элемент имеет пустое содержание.

1.5. Блочные элементы

Помимо абзацев, задаваемых элементом `p`, содержимое страницы может включать заголовки (не путать с названием страницы, задаваемым элементом `title`), задаваемые элементами `h1`, `h2`, ..., `h6`, причем элемент `h1` задает заголовок первого уровня, `h2` – второго и так далее. Заголовки являются блочными элементами, а их содержание подчиняется тем же правилам, что и содержание элемента `p`.

Еще один вид блочных элементов – это `blockquote`. Этот элемент задает блочную цитату. В отличие от элемента `q` элемент `blockquote` должен содержать один или несколько блочных элементов. Обычно браузеры выделяют цитаты отступом.

Кроме абзацев, блочных цитат и заголовков еще одним видом блочных элементов являются списки.

Поддерживается три вида списков:

- нумерованный,
- маркированный,
- список определений.

Нумерованный и маркированный список задаются соответственно элементами `ol` и `ul`. Эти элементы должны содержать один или несколько элементов `li`, задающих каждый отдельный элемент списка.

В свою очередь содержимым элементов `li` может быть как обычный текст, так и один или несколько блочных или строчных элементов.

Поскольку списки сами являются блочными элементами, и, стало быть, могут включаться в элемент `li`, возникает возможность создавать многоуровневые списки

Например:

```
<ol>
  <li>
    Visual Basic for Applications
    <ol>
      <li>Типы данных</li>
      <li>Встроенные функции</li>
      <li>Массивы</li>
    </ol>
  </li>
  <li>HTML</li>
</ol>
```

1.6. Атрибуты

В языке HTML элементы могут иметь не только имя, но и набор атрибутов. Атрибут состоит из имени и значения. Атрибуты перечисляются в открывающем теге. Приведем пример:

```
<p lang="ru">Текст по-русски</p>
```

Здесь элемент `p` снабжен одним атрибутом `lang`, значением которого является `ru`. Каждый атрибут может встречаться в теге не более одного раза; порядок перечисления значения не имеет. Значение атрибута можно записывать как в двойных, так и в одинарных кавычках. Если кавычки встречаются в самом значении атрибута, то их следует удвоить.

Набор допустимых атрибутов, а также множество допустимых значений каждого из атрибутов зависит от имени элемента. Так, использованный в данном примере атрибут `lang` задает язык; его значением должен быть код языка, например: `ru`, `en`, `fr`, `de`, `jp` и так далее. Этот атрибут может применяться ко всем элементам, в которых (прямо или косвенно) может содержаться текст, отображаемый на экране. Обычно атрибут `lang` применяется к элементу `html`.

1.7. Гиперссылки

Гиперссылка – это фрагмент страницы, щелкнув на котором окне браузера пользователь может перейти к другой странице или же к другой части данной страницы. Обычно браузер выделяет ссылки подсветкой и подчеркиванием. Гиперссылка является строчным элементом и задается элементом `a`.
Например:

```
<a href="more-info.html">Дополнительная  
информация</a>
```

Адрес перехода задается атрибутом `href`; этот адрес может указывать как на веб-страницу, так и на ресурс любого другого типа. Адрес должен иметь форму универсального идентификатора ресурса (URI – Universe Resource Identifier) и может быть как абсолютным, так и относительным. В приведенном примере адрес является относительным.

Это означает, что, если, например, страница, содержащая эту ссылку, находится по адресу <http://www.example.ru/data/index.html>, то, активизировав ее, пользователь перейдет на страницу, находящуюся по адресу <http://www.example.ru/data/more-info.html>.

Кроме перехода к другому ресурсу гиперссылку можно использовать для перехода к другой части данной страницы.

Для этого соответствующую часть снабдить идентификатором, задаваемым атрибутом `id`, например:

```
<p id="more-info">Здесь размещается дополнительная информация</p>
```


1.8. Изображения

Для вставки изображений используется строчный элемент `img`. Адрес, по которому находится графическое изображение, задается атрибутом `src`. Как и в случае атрибута `href` адрес может быть как абсолютным, так и относительным.

Обычно используются изображения в формате GIF, JPEG или PNG.

Атрибут `src` является обязательным.

Другой обязательный атрибут — `alt` — задает текст, который будет выводиться браузером вместо графического изображения в том случае, если загрузка последнего еще не завершена либо не удалась.

Кроме обязательных атрибутов `src` и `alt` к элементу `img` можно применить атрибуты `width` и `height`, задающие размер изображения в пикселях или в процентах относительно размера включающего блока.

Основное назначение атрибутов `width` и `height` — не изменение размеров (этого не следует делать из-за возможности снижения качества), а возможность сообщить браузеру размеры изображения до фактической загрузки самого изображения. Элемент `img` не имеет содержания.

Приведем пример:

```

```

1.9. Таблицы

Кроме вышеперечисленных блочных элементов существует еще один – элемент `table`. Этот элемент задает таблицу. Приведем пример простейшей

```
<table rules="all">
  <tbody>
    <tr><th>Страна</th><th>Столица</th></tr>
    <tr><td>Россия</td><td>Москва</td></tr>
    <tr><td>Великобритания</td><td>Лондон</td></tr>
    <tr><td>Германия</td><td>Берлин</td></tr>
  </tbody>
</table>
```

Элемент `table` должен содержать в себе, по крайней мере, один элемент `tbody`. Элемент `tbody` представляет группу строк таблицы и должен включать один или несколько элементов `tr`, представляющих отдельные строки, которые, в свою очередь, должны включать один или несколько элементов `td` или `th`.

Элементы `td` и `th` представляют отдельные ячейки таблицы и могут содержать как обычный текст, так и строчные и блочные элементы. Элемент `th` отличается от `td` тем, что задает ячейку, используемую в качестве заголовка строки или столбца. Атрибут `rules` задает разделительные линии.

Возможные значения этого атрибута `rules` :

- `none` (без разделительных линий),
- `groups` (разделительные линии между группами строк и столбцов),
- `rows` (разделительные линии между отдельными строками),
- `cols` (разделительные линии между отдельными столбцами),
- `all` (все разделительные линии).

К элементам `td` и `th` можно применять атрибуты **`rowspan`** и **`colspan`**, которые позволяют растянуть соответствующую ячейку на указанное количество строк и столбцов.

В элементе `table` элементам `tbody` может предшествовать по одному элементу `thead` и `tfoot`. Эти элементы задают соответственно заголовок и подвал таблицы. Их содержание аналогично элементу `tbody`.

Для задания параметров колонок могут использоваться элементы `col` и `colgroup`, включаемые в элемент `table` перед элементами `tbody`, `thead` и `tfoot`. К элементу `col` можно применить атрибут `width`, задающий ширину колонки.

Ширина может задаваться как в пикселях, так и в процентах от ширины всей таблицы. Кроме этого ширина может быть относительной, заданной долей ширины таблицы за вычетом общей ширины тех столбцов, ширина которых задана в пикселях или процентах.

Также к элементу `col` можно применить атрибут `span`, задающий количество столбцов, на которые воздействует данный элемент. Элемент `colgroup` позволяет сгруппировать несколько элементов `col`. К элементу `colgroup` применимы те же атрибуты, что и к элементу `col`.

Ко всем вышеперечисленным элементам, используемым для создания таблиц, также применимы следующие атрибуты выравнивания:

- `align` – горизонтальное выравнивание. Возможные значения: `left` (по левому краю), `center` (по центру), `right` (по правому краю), `justify` (по обоим краям) и `char` (по указанному символу);
- `char` – символ для выравнивания (используется, если значением атрибута `align` является `char`);
- `valign` – вертикальное выравнивание. Возможные значения: `top` (по верхнему краю), `middle` (по центру), `bottom` (по нижнему краю) и `baseline` (по базовой линии).

Наконец, таблица может иметь название. Оно задается элементом `caption`, содержимым которого может быть обычный текст и строчные элементы. Элемент `caption` размещается в элементе `table` перед всеми остальными элементами.

Пример таблицы:

```
<table summary="Объединение ячеек">
  <tbody>
    <tr>
      <td rowspan="3">1</td>
      <td colspan="2">2</td>
      <td rowspan="3">3</td>
    </tr>
    <tr>
      <td>4</td>
      <td>5</td>
    </tr>
    <tr>
      <td colspan="2">6</td>
    </tr>
  </tbody>
</table>
```

1.10. Формы

По своей структуре данные, отправляемые формой, представляют собой набор свойств, каждое из которых состоит из имени и значения, причем одни и те же имена свойств могут повторяться в комбинации с разными значениями.

В коде форма представляет собой элемент `form`, размещаемый в пределах тела страницы и содержащий элементы управления (поля ввода, кнопки, списки и т. п.) вперемешку с обычными элементами разметки. Одна страница может содержать несколько форм.

У элемента form могут быть следующие атрибуты:

- action — адрес, на который будут отправлены данные, введенные в элементы управления формой. Предполагается, что по этому адресу располагается серверный сценарий, который сможет как-либо обработать данные формы (например, эти данные могут быть занесены в базу данных). По умолчанию данные отправляются на тот же адрес, откуда была получена страница, содержащая форму. У адреса, заданного атрибутом action, может быть схема отличная от http и https. Например, если используется схема mailto, то данные формы отправляются на заданный адрес электронной почты;

- `method` – способ отправки данных. Возможными значениями этого атрибута являются ключевые слова `get` и `post`. При использовании в атрибуте `action` схем `http` и `https` атрибут `method` задает используемый метод протокола HTTP, а при использовании схемы `mailto` способ размещения данных в почтовом сообщении. Если для отправки данных используется метод `GET`, то данные формы подставляются в строку запроса и становятся видны на экране браузера в строке адреса; если же используется метод `POST`, то отправляемые данные формы образуют тело запроса и на экране не показываются. Также необходимо принимать во внимание следующее: браузер считает, что выполнение метода `POST` может изменить состояние сервера и поэтому каждый раз при попытке его выполнения требует от пользователя явного подтверждения;

- `enctype` — способ кодирования данных формы.

Возможные значения этого атрибута: `application/x-www-form-urlencoded` (это значение по умолчанию) и `multipart/form-data`. Кодирование `multipart/form-data` используют тогда, когда форма предназначена для отправки файлов (т. е. на форме есть элемент управления, позволяющий выбрать файл, либо присоединение файлов реализовано программно с использованием сценария на языке JavaScript).

Простейшие элементы управления (поля ввода, кнопки и переключатели) представляются элементом `input`. В данных формы каждому простейшему элементу может соответствовать только одно свойство.

Атрибуты элемента `input`:

- `name` — имя элемента — задает имя свойства, под которым значение данного компонента будет представлено в данных формы. Если этот атрибут опустить, то данные, введенные в элемент управления, отправлены не будут, хотя сам он будет присутствовать на экране;
- `value` — значение элемента — задает значение соответствующего свойства в данных формы. Если элемент управления дает возможность пользователю редактировать содержащееся в нем значение, то данный атрибут (если он задан) интерпретируется как значение по умолчанию;
- `type` — тип элемента управления. Для задания используются следующие константы:
 - `text` — поле для ввода текста;
 - `password` — как предыдущее, но вместо вводимого текста отображаются звездочки или точки. Используется для ввода паролей;
 - `checkbox` — независимый переключатель (обычно в виде квадратика, в котором можно поставить галочку);

- radio — зависимый переключатель (обычно в виде кружка, в котором можно поставить точку). От независимого отличается тем, что в каждый момент времени из всех имеющихся на одной форме переключателей с одинаковым значением атрибута name включенным может быть только один (переключатели с разным значением атрибута name могут быть включены одновременно);
- submit — кнопка, при щелчке на которой происходит отправка формы. По умолчанию после отправки формы браузер переходит по адресу, заданному в атрибуте action. Надпись на кнопке задается атрибутом value.
- reset — кнопка, при щелчке на которой происходит сброс значений, введенных в элементы управления формы;
- button — кнопка без predetermined действия. Для задания действий используется сценарий на языке JavaScript;
- file — поле для выбора файла, содержимое которого будет отправлено в составе данных формы. Отправка файла возможна только в том случае, когда для формы задана кодирование multipart/form-data;

`hidden` — скрытое поле. Поле данного типа не отображается на экране. Используется для того, чтобы связать с формой дополнительные данные;

· `size` — размер поля для ввода, измеренный в символах. Этот параметр не ограничивает количества символов, которое можно ввести в это поле;
`maxlength` — максимальное количество символов, которое можно ввести в поле ввода;

`checked` — является ли переключатель по умолчанию включенным. Этот атрибут является логическим и записывается без значения.

У элемента `input` не должно быть содержания.

Обычно элементы управления, будучи расположенными на странице, сопровождаются метками. Для задания меток следует использовать элемент `label`. Содержимым этого элемента может быть произвольный текст, причем, если среди этого текста встречается элемент управления, то по умолчанию метка будет относиться именно к нему. Также помечаемый элемент управления можно задать явно, воспользовавшись атрибутом `for`, значением которого должен быть идентификатор элемента управления.

Кроме элемента `input` для создания элементов управления можно использовать и некоторые другие элементы.

Для создания многострочного поля ввода используется элемент `textarea`. Этот элемент поддерживает следующие атрибуты:

- `name` и `maxlength` — аналогично одноименным атрибутам элемента `input`;
- `cols` — ширина поля ввода, измеренная в символах (аналогично атрибуту `size` элемента `input`);
- `rows` — высота поля ввода, измеренная в строках;
- `wrap` — будет ли при отправке данных сохранено автоматическое разбиение на строки. Возможные значения этого атрибута представлены ключевыми словам `hard` (сохраняется) и `soft` (не сохраняется). По умолчанию автоматическое разбиение на строки не сохраняется. Отметим, что ручное разбиение на строки сохраняется любом случае.

Для создания списка с возможностью выбора одной или нескольких опций используется элемент `select`, поддерживающий следующие атрибуты:

- `name` — аналогично одноименному атрибуту элементов
- `input` и `textarea`;
- `size` — количество одновременно отображаемых опций. Если этот атрибут равен единице или отсутствует, то создается раскрывающийся список. В противном случае создается обычный список. Если значение атрибута `size` меньше фактического количества опций, то список отображается с вертикальной полосой прокрутки;
- `multiple` — допускается ли выбор нескольких опций. Этот атрибут является логическим; если он установлен, то вне зависимости от атрибута `size` создается обычный (а не раскрывающийся) список. Обычно для выбора нескольких элементов используется клавиша `Ctrl`.

Содержимым элемента `select` должны быть опции и группы опций, задаваемые соответственно элементами `option` и `optgroup`. Элемент `option` поддерживает следующие атрибуты:

- `value` — значение свойства данных формы в случае выбора данной опции;
- `label` — метка, представляющая данную опцию на экране. Если этот атрибут отсутствует, то используется содержание элемента;
- `selected` — является ли данная опция выбранной по умолчанию. Этот атрибут является логическим.

Элемент `optgroup` поддерживает атрибут `label`, аналогичный одноименному атрибуту элемента `option`. Содержимым элемента `optgroup` должны быть только элементы `option`, т. е. вложенные группы не поддерживаются.

Для создания кнопки (помимо элемента `input` с атрибутом `type`, равным `button`) можно использовать элемент `button`, отличие которого заключается в том, что надпись, отображаемая на кнопке, задается содержимым элемента и может не совпадать со значением, задаваемым атрибутом `value`. Кнопка поддерживает атрибут `type`, которые может принимать значения `submit`, `reset` и `button`.

Приведем пример:

```
<form action="add-comment.php" method="post">
  <label class="row">
    <span class="col1">Text</span>
    <span class="col2">
      <textarea name="text"></textarea>
    </span>
  </label>
  <label class="row">
    <span class="col1">Rating</span>
    <span class="col2">
      <select name="rating">
        <option value="excellent" selected>отлично</option>
        <option value="good">хорошо</option>
        <option value="fair">посредственно</option>
        <option value="poor">плохо</option>
      </select>
    </span>
  </label>
  <button type="submit">Save</button>
</form>
```

1.11. Фреймы

Фрейм — это блок в составе одной страницы, в котором отображается содержание другой страницы. Фрейм задается элементом `iframe`, у которого могут быть следующие атрибуты:

`src` — адрес страницы, отображаемой во фрейме;

`name` — имя фрейма;

`width` и `height` — размеры фрейма с пикселях.

Пример:

```
<iframe width="560" height="315"  
src="http://www.youtube.com/embed/Q39NNcc81P4"></iframe  
>
```

Контрольные вопросы

1. Что такое элемент? В чем разница между элементом и тэгом?
2. Что такое атрибут элемента?
3. Какова структура веб-страницы?
4. Что может находиться в заголовке (head) веб-страницы?
5. Что может находиться в теле (body) веб-страницы?
6. Что такое комментарий и для чего он используется?
7. Что такое сущность? Какие виды сущностей поддерживаются?
8. В чем разница между блочным и строчными элементами?
9. Какие виды списков поддерживаются в HTML?
10. Что такое гиперссылка?
11. Как вставить графическое изображение? Какие форматы графических изображений поддерживаются современными браузерами?
12. Как вставить таблицу? Какие структурные элементы таблицы поддерживаются в HTML?
13. Для чего используются формы?
14. Какие элементы управления поддерживаются в HTML?
15. Для чего используются фреймы?