Занятие 1. Основы HTML

Введение

Доброго времени суток, читатель! Это текстовый вариант 1 урока из курса начинающего фронтенд разработчика.

Возможно, Вы незнакомы с понятие **фронтенд**? Фронтенд (eng. *Front-end*) - это та часть программного обеспечения, которая работает на стороне клиента (т.е. на вашем устройстве) и обменивается данными с сервером, который обычно называют **бэкенд** (eng. *Back-end*). Все достаточно просто.

Современный фронтенд-разработчик сталкивается с довольно разнообразным спектром задач, так что скучать не приходится. Начиная от разработки пользовательских **интерфейсов**, до написания и внедрения **скриптов**, отвечающих за логику в веб-приложениях.

Но пока это все вдалеке от нас. Сейчас мы с вами только в самом начале пути, ведущем в мир фронтенда.
И как очевидно бы это не звучало, начнем мы познавать этот мир с самых основ. А именно с **HTML**.

HTML или Язык гипертекстовой разметки

HTML (eng. *Hypertext markup language*) - это язык гипертекстовой разметки, который позволяет описать структуру веб-страниц и её **контента** - содержимого старницы.

Если говорить проще, то задача языка HTML - разбиение тестовых данных в понятные для браузера блоки, которые можно связать с уже существующей информацией при помощи гиперссылок.

Для того, чтобы начать использовать HTML достаточно просто открыть блокнот и сохранить файл с расширением `.html`. После сохранения, иконка файла сменится на иконку вашего браузера, установленного по умолчанию.

В жизни для редактирования HTML документов вместо блокнота обычно используют один из существующих редакторов кода, вроде Atom, Sublime Text 3 или Visual Studio Code.

HTML элементы

Язык HTML представлен множеством элементов, каждый из которых имеет **семантику** (проще говоря, смысл) и некоторый базовый вид в браузере.

К примеру элемент параграф (eng. *Paragraph*):

```
<p>Мой кот очень пушистый</p>
```

Он состоит из открывающего и закрывающего **тегов** (`<p>` и `</p>`, соответственно), а также контента `Мой кот очень пушистый`.

Вместе они образуют **HTML-элемент**, в данном случае элемент параграф, который поясняет браузеру, что это абзац и его нужно отделить небольшими отступами сверху и снизу.

Большинство тегов, доступных в языке являются парными, как `<p>`. Такие теги требуют закрывающей пары, которая отличается лишь наличием косой черты, вроде `</p>`.

Базовая структура HTML документа

> Для лучшего понимания рекомендую создать на вашем локальном диске папку MyProject, а в ней создать файл page.html и самостоятельно писать и запускать приведенные примеры кода. Это ускорит процесс обучения и гораздо интересней, чем простое чтение

Попробуем разобраться со структурой рядового HTML документа, который обычно и называют **веб-страницей**.

```
<!DOCTYPE html>
<html>
  <head>
    <title>О моем коте</title>
  </head>
  <body>
    <p>Мой кот очень пушистый.</p>
    <p>Он любит есть.</p>
  </body>
</html>
```

Первая строчка любой веб-страницы это DOCTYPE - формат документа:

```
<!DOCTYPE html>
```

Он указывает браузеру на то, что документ состоит из HTML тегов и их не нужно выводить на страницу.

Сразу за ним следуют парные теги `<html>`

```
<html>
...
</html>
```

которые определяют элемент `html` - область, в которой необходимо писать HTML-код. Часто этот элемент называют **корневым** элементом веб-страницы.

Весь код, который мы пишем на HTML можно разбить на две основные области. Первая - это элемент `head`.

```
<head>
  <title>О моем коте</title>
</head>
```

Этот элемент является контейнером для всего, что будет использоваться на странице, но не будет отображено на прямую, как контент. Сюда относятся файлы стилей и скрипты, о которых мы поговорим позднее, разного рода информация, вроде кодировки, ключевые слова по странице и прочее.

В нашей ситуации здесь содержится только элемент `title`, который устанавливает название вкладки в браузере, в панели открытых вкладок.

Второй областью является элемент `body`, который называется **телом страницы**.

```
<body>
  <p>Мой кот очень пушистый.</p>
  <p>Он любит есть.</p>
</body>
```

Именно эта зона будет отображена в браузере, когда вы откроете страницу. Всю информацию, которую вы хотите отобразить на странице необходимо помещать между тегами `<body>...</body>`.

Желая поделиться с миром информацией о своем коте, вы вероятно, напишите о нем парочку-другую строк.

Вот и сейчас мы поместили в `body` немного про пушистого:

```
<p>Мой кот очень пушистый.</p>
<p>Он любит есть.</p>
```

А вообще у меня не пушистый кот, а толстый...

Текстовое форматирование

Думаю, уже пора перейти к одной из основных возможностей HTML - текстовому форматированию.

В HTML достаточно много элементов, который позволяют объяснить браузеру, что же содержится на странице.

Один из самых популярных элементов в этой группе мы уже рассмотрели - это элемент `p`.

> Подразумевается, что все что мы пишем далее находится между тегами `<body></body>`. Современные браузеры поправят вас, если вы поместите информацию вне этих тегов, однако такая разметка будет считаться **невалидной**, т.е. некорректной с точки зрения правил языка.

Заголовки

Один из самых простых способов сделать текст удобным для чтения, помимо выделения абзацев, является его разделение на смысловые части, которым предшествуют заголовки.

```
<h1>О мой кот!</h1>
```

Заголовки разделяют по степени важности (перечислены по убыванию):

```
<h1>О мой кот!</h1>
...
<h2>Тоже мой, только поменьше</h2>
...
<h3>Средних размеров кот</h3>
...
<h4>Кот "Четвертинка"</h4>
...
<h5>Небольшой Кот</h5>
...
<h6>Кот, которого почти никто не замечает</h6>
```

Здесь `h1` - самый главный заголовок, обычно являющийся названием статьи или определяющий смысл дальнейшего текста, а `h6` заголовок, выделяющий небольшой подраздел в основном тексте. В общем случае, заголовки одного типа могут повторяться на странице.

Правилом хорошего тона является отсутствие пропусков в номерах заголовков. То есть вместо набора из `h1`, `h4` и `h6`, стоит делать выбор в пользу комбинации `h1`, `h2` и `h3`.
Это создаёт грамотную структуру представляемого контента, которую принято называть **outline** (eng. *Outline* - схема, набросок).

Списки

При написании докладов, статей и прочей текстовой информации достаточно часто возникает потребность собрать какие-то мысли в список, перечислить их по порядку. В HTML для этих целей используются 3 типа списков:

1. Несортированный список (eng. *Unsorted list*)

```
<ul>
  <li>Кот Белый</li>
  <li>Кот Черный</li>
  <li>Кот "Рыжик"</li>
</ul>
```

Для того, чтобы добавить пункт в список необходимо использовать элемент `li` (eng. *List item*). Каждый элемент данного списка имеет маркер - черный кружок слева от пункта.

2. Упорядоченный список (eng. *Ordered list*)

```
<ol>
  <li>Кот Белый</li>
  <li>Кот Черный</li>
  <li>Кот "Рыжик"</li>
</ol>
```

Упорядоченный отличается от несортированного наличием нумерации каждого элемента списка (от 1 и по порядку).

3. Список определений (eng. *Description list*)

```
<dl>
  <dt>Кот Белый</dt>
  <dd>
    Традиционный дизайн почти всех классов котов.
    Имеется существенный недостаток - на белом цвете
    сильно заметны загрязнения и, как следствие,
    необходимо часто проводить процедуру отчистки.
  </dd>
  <dt>Кот Черный</dt>
  <dd>
    Готический дизайн, характерный для Пражских котов.
    Когда-то имел скверную репутацию, поскольку считалось,
    что если черный кот перейдет дорогу, то будут
    неприятности. Недостатком является неуловимость
    данной разновидности во тьме ночной.
  </dd>
  <dt>Кот "Рыжик"</dt>
  <dd>...</dd>
</dl>
```

Как видите, список определений несколько отличается от двух предыдущих списков. Данный тип тоже достаточно простой: список состоит из пар “Термин” - “Определение”.

Термину соответствует элемент `dt` (eng. *Definition term*). В свою очередь определение записывается внутри элемента `dd` (eng. *Definition description*).

Цитаты

> Не грех процитировать умную мысль, даже если она принадлежит тебе.

Для того, чтобы добавить цитату на страницу достаточно использовать элемент `blockquote`:

```
<p>Марк Твен однажды сказал:</p>
<blockquote cite="ru.wikiquote.org/wiki/Марк_Твен">
  "Ничто так не нуждается в исправлении, как чужие привычки."
</blockquote>
```

Такой вид цитаты является **блочным**, т.е. отделяется от остальной части контента. Если же вы хотите добавить цитату прямо как **строчный** текст, то следует использовать элемент `q`.

```
<p>Марк Твен однажды сказал: <q cite="ru.wikiquote.org/wiki/Марк_Твен">
   Ничто так не нуждается в исправлении, как чужие привычки.</q></p>
```

При использовании строчного элемента `q` (eng. *Quote*), нет необходимости писать кавычки, поскольку браузер добавит их автоматически.

“Важные” теги

Кажется каждый хоть раз сталкивался с рекламой содержащей броскую фразу, вроде “**Купи кота, а то потом не будет**”, написанной большим и жирным шрифтом. Задача такой рекламы, очевидно, обратить на себя внимание. В тексте эта задача - достаточно частое явления.

HTML предоставляет ряд элементов, которые могут использоваться с целью усиления значения определенных фраз или привлечения внимания читателя к ним. К категории “читатель” можно также отнести и поисковые системы, поскольку эти системы уделяют некоторое внимание выделенным фразам, особенно если они являются **ключевыми**.

Раз уж мы затронули вопрос привлечения внимания, то давайте рассмотрим элемент `em` (eng. *Emphasize*), в чьи обязанности входит создание контраста или акцента на указанном тексте. По умолчанию браузеры отображают содержание этого элемента курсивом.

```
<p><em>Насколько</em> ваш кот пушистый?</p>
```

— *Насколько* ваш кот пушистый?

Однако помимо `em` браузеры выделяют также выделяют курсивом и элемент `i` (eng. *Italic*).Разница лишь в том, что `i` не имеет никакой семантики (смысла) и используется обычно там, где нужно визуально изменить текст, который чем-то отличается от остального. Например, при использовании названия книги в абзаце.

```
<p><i>Насколько</i> ваш кот пушистый?</p>
```

— *Насколько* ваш кот пушистый?

Далее идет `strong`. Он подчеркивает важность выделенной информации и по умолчанию отображается в браузере, как текст с жирным начертанием.

```
<p>Мой кот <strong>очень</strong> пушистый.</p>
```

— Мой кот **очень** пушистый.

По мимо элемента `strong` имеется еще `b` (eng. *Bold*), который также позволяет отобразить текст жирным, однако он (как и `i`) не имеет никакой семантики и обычно используется в презентационных целях.

```
<p>Мой кот <b>очень</b> пушистый.</p>
```

— Мой кот **очень** пушистый.

Все эти теги (как в общем-то и другие в языке) можно комбинировать друг с другом и получать одновременно жирный и наклонный текст.

```
<p>Ваш кот <strong><i>достаточно</i></strong> пушистый?</p>
```

— Ваш кот ***достаточно*** пушистый?

Порядок для комбинирования в данном случае не важен, главное соблюдайте правильно уровни вложенности тегов, то есть тот тег, который открылся первым, должен закрыться последним.

Атрибуты HTML-элементов

Думаю, вы заметили, что внутри открывающего тега `<blockquote>` присутствовал **атрибут** `cite="..."`. Атрибуты - это дополнительные параметры, которые настраивают элементы, регулируют их поведение различным способом или позволяют передать важную для браузера информацию о элементе.

В данной ситуации атрибут `cite` тега `<blockquote>` сообщает браузеру адрес ресурса, которая является источником цитаты - ru.wikiquote.org/wiki/Марк_Твен.

Все атрибуты располагаются внутри открывающего тега, сразу после его имени в формате:

```
<тег атрибут="значение" атрибут_2="значение_2" ...>
```

Тег и атрибуты (в том числе между собой) разделяются пробелом, перед закрывающим знаком `>` пробел необязателен.

Существует множество атрибутов, среди который 16 являются **глобальными** (общими для всех элементов). Подробнее атрибуты мы разберем позднее.

Изображения

Веб - это огромная база знаний, в которой содержится невероятно количество текстовой информации (преимущественно о котах, конечно же). Однако обойтись лишь текстовой информацией трудно. Человек по своей природе привык усваивать всё с дополнительным визуальным контентом. Так почему бы не добавить чуточку изображений? Котов, конечно же.

```
<img src="http://random.cat/i/UwL2m.jpg" alt="I'm a cute kitty!">
```

Заметьте, что тег `<img>` не является парным, а значит не имеет закрывающего тега.

Атрибут `src` является главным для элемента, поскольку он определяет расположение интересующего нас изображения.
В отличие от `cite`, который важен для браузера и поисковых систем, но не для самого пользователя, `img` влияет на содержимое страницы: без него браузер бы не знал, откуда загружать фотографию этого прелестного создания.

В свою очередь атрибут `alt` тоже оказывается достаточно значимым, поскольку заменяет изображение, которое по какой-то причине не может быть отображено, текстом-подсказкой, а также напрямую влияет на **доступность**, так как программы для чтения текста с экрана (eng. *Screen readers*) озвучивают именно значение этого атрибута.