

Разметка документов

Как правильно разметить содержимое?

Для того, чтобы правильно разметить содержимое необходимо использовать *семантическую разметку*.

Семантическая разметка задается с помощью тегов, т.е. Вы используете заголовки, теги абзацев и т.д.

Plain Old Semantic HTML (POSH)

Семантическая разметка требует избегать таких элементов, как `font` и `table`.

Семантическая разметка (POSH) полностью фокусируется на контенте. Основная суть - использовать элемент, который лучше всего опишет контент.

Однако, чтобы создавать ВАЛИДНЫЙ код необходимо знать, что некоторые HTML элементы в 5 версии устарели.

Устаревшие элементы

Элемент

- Basefont
- Big
- Center
- Font
- paintext
- S
- Striike
- Tt
- U
- I
- B
- Frame
- Frameset
- noframes

Атрибут

- Align
- Background
- Bgcolor
- Border
- Cellpadding
- Cellspacing
- Height
- Nowran

Выбор синтаксического стиля кода

- Необходимо учитывать верхний регистр в названии тегов
- Необязательны кавычки для атрибутов
- Необязательны значения для атрибутов
- Необязательны закрывающие теги для пустых элементов

Однако, это не обязательно!

Микроразметка

Разметка, увеличивающая значимость контента - важный инструмент современного веб-разработчика.

Существует несколько видов различной микроразметки, однако, большинство поисковых систем отдает предпочтение стандартам **Schema.org**

Микроразметка

Для того, чтобы добавить данные микроразметки используют атрибуты `itemscope` и `itemprop`.

```
<p itemscope>  
<span itemprop="изобретатель">Тим ВерНерс-/n</span> создал  
<span itemprop="изобретение">Всемирную паутину<Лрап>  
</p>
```

Микроразметка

Itemscope применяется для определения области действий микроданных, т.е. набора - имя/значение.

Значение **itemprop** устанавливает имена свойств и связанную с ним информацию.

Имеем следующие пары имя/значение:

Изобретатель/Тим Бернерс-Ли

Изобретение/Всемирная паутина

Микроразметка

При использовании `schema.org` необходимо не только указать дополнительный атрибут `itemtype`, но еще и назначать имена исключительно в соответствии с лексикой `Schema.org`

```
<section itemscope itemtype="http://schema.org/Person">  
<h1 itemprop="name">ТНМ Бернере-Ли</h1>  
  
</section>
```

Микроразметка

<http://ruschema.org/docs/schemas> - подробная информация о видах разметки schema.org

Доступность

- *Восприимчивость* — означает, что информация должна быть доступной для восприятия, как правило, зрительно или на слух. Необходимо предоставить альтернативный текст для изображений, заголовки для аудио и видео, а также обеспечить достаточный контраст между текстом и фоном

Доступность

- *Работоспособность* – гарантирует, что посетители смогут взаимодействовать между собой и пользоваться сайтом через навигацию, контент, формы и динамический контроль.

Доступность

- *Понятность* — означает, что текст должен быть четким, а поведение сайта — последовательным и предсказуемым.

Доступность

- *Надежность* – определяет разметку страницы и программирует работу с различными пользовательскими агентами и вспомогательными технологиями.

Работа с мультимедиа

Работа с мультимедиа. Графика

Впервые изображение в html-файлах стало возможно добавлять в 1997 году.

Современные браузеры поддерживают следующие форматы растровых изображений:

- *jpg*,
- *gif*,
- *bmp*,
- *tiff*,
- *png*,
- *psd*

Работа с мультимедиа. Графика

Также браузеры поддерживают векторную графику. Такие форматы могут отображаться:

- *swf*,
- *cdr*,
- *max*,
- *ai*

Работа с мультимедиа. Графика.

Структура мультимедийной информации принципиально отличается от структуры текстовой, поэтому напрямую мультимедиа не может быть описана в *html*-коде. Вся необходимая разработчику мультимедиа содержится в отдельных файлах, ссылки на которые, в виде соответствующих тегов прописываются в *html*-коде.

Работа с мультимедиа. Графика.

Для добавления изображения на страницу используется тег ``, который содержит **обязательный атрибут `src`**, в котором указан путь к файлу.

```

```

Работа с мультимедиа. Графика

К числу необязательных атрибутов `` 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.

Работа с мультимедиа. Аудио и видео

HTML5 не используют никаких дополнительных инструментов для воспроизведения аудио или видео.

Это повышает безопасность, увеличивает интеграцию, позволяет задействовать меньше аппаратных ресурсов.

Работа с мультимедиа. Аудио и видео

В качестве основных недостатков - поддержка далеко не всех кодеков.

Работа с мультимедиа Аудио и видео

Для работы с аудио HTML5 использует одноименный тег <audio>.

```
<audio src="1.mp3"></audio>
```

Работа с мультимедиа. Аудио и видео

- Тег `<audio>` содержит следующие атрибуты:
- `autoplay` - при его добавлении, воспроизведение файла начинается сразу же после загрузки страницы;
- `controls` - добавляет панель управления к аудио;
- `loop` - воспроизведение аудио повторяется с начала, после его завершения;
- `preload` - используется для загрузки файла вместе с загрузкой самой страницы, игнорируется, если использован `autoplay`;
- `src` - задает путь к файлу для воспроизведения.

Работа с мультимедиа. Аудио и видео

Для работы с видео используется одноименный элемент `<video>`

```
<video src="2.mp4"></video>
```

Работа с мультимедиа. Аудио и видео

- Атрибуты тега `<video>`:
- `autoplay` - при его добавлении, воспроизведение файла начинается сразу же после загрузки страницы;
- `controls` - добавляет панель управления к видео;
- `height` - задает высоту области, для воспроизведения видео.
- `loop` - воспроизведение аудио повторяется с начала, после его завершения;
- `poster` - указывает путь к изображению, пока видео не воспроизводится, или недоступно;
- `preload` - используется для загрузки видеофайла вместе с загрузкой самой страницы, игнорируется, если использован *autoplay*;
- `src` - задает путь к файлу для воспроизведения.
- `width` - задает ширину области, для воспроизведения видео.

Работа с мультимедиа. Аудио и видео

Спецификацией HTML5 не поддерживаются следующие возможности элементов `<audio>` и `<video>`:

- Воспроизведение потокового мультимедиа. В настоящий момент есть только приложения, предусматривающие поддержку воспроизведения потоковой мультимедиа.

Потоковое воспроизведение - это непрерывное воспроизведение.

Работа с мультимедиа. Аудио и видео

- Ограничения кроссдоменного разделения ресурсов (CORS).
- Технология разделения ресурсов предусматривает предоставление веб-странице доступ к ресурсам другого домена.

Работа с мультимедиа. Аудио и видео

- Невозможность воспроизведения из сценариев полноэкранного видео, из-за соображений обеспечения безопасности. Как правило, это ограничение компенсируется предоставлением дополнительных элементов управления браузера.
- Отсутствие спецификации доступности элементов `<audio>` и `<video>` для людей с ограниченными возможностями. Создается спецификация WebSTR, которая должна регламентировать поддержку субтитров формата STR.

Canvas

Canvas

- Благодаря спецификации языка HTML5 появился элемент `canvas`, благодаря которому появляется мощный инструментарий для рисования с использованием JavaScript.
- Для каждого элемента `canvas` можно использовать "контекст" (представьте страницу в альбоме для рисования), в который можно выполнять команды JavaScript для рисования. Браузеры могут реализовать несколько контекстов холстов и различные API предоставляют функции для рисования.

Canvas

- В основном реализация в современных браузерах производится в 2D-функциях холста.

Canvas

Создание холста доступно с помощью теги `<canvas>`

```
<canvas id="myCanvas" width="200" height="150"> </canvas>
```

Элементу `canvas` необходимо присваивать `id`, потому что в дальнейшем обращение в JS будет происходить по `id`.

Canvas

- Создан
рисуван

```
window.addEventListener('load', function ()
{ // Получаем ссылку на элемент.
var elem = document.getElementById('myCanvas');
if (!elem || !elem.getContext) { return; }
// Получаем контекст 2d.
var context = elem.getContext('2d');
if (!context) { return; }
// Все сделано! Теперь можно нарисовать синий
прямоугольник.
context.fillStyle = '#00f';
context.fillRect(0, 0, 150, 100); }, false);
```

Canvas

```
context.fillStyle = '#00f';  
// синий  
context.strokeStyle = '#f00';  
// красный  
context.lineWidth = 4;  
// Рисуем несколько прямоугольников.  
context.fillRect (0, 0, 150, 50);  
context.strokeRect(0, 60, 150, 50);  
context.clearRect (30, 25, 90, 60);  
context.strokeRect(30, 25, 90, 60);
```

Функции `StrokeStyle` `FillStyle` - позволяет назначить для будущей фигуры стиль

Функция `LineWidth` - прорисовка толщины линий

Функция `fillRect`, `strokeRect` позволяют нарисовать прямоугольник. В скобках указаны координаты, а также его ширина и длина.

Функция `clearRect` очищает прямоугольную область

Canvas

```
// Задаем свойства стиля оформления.  
context.fillStyle = '#00f';  
context.strokeStyle = '#f00';  
context.lineWidth = 4;  
context.beginPath();  
// Начинаем с верхней левой точки.  
context.moveTo(10, 10);  
// задаем координаты (x,y)  
context.lineTo(100, 10);  
context.lineTo(10, 100);  
context.lineTo(10, 10);  
// Готово! Теперь заполните фигуру, и начертите штрихи. //  
Примечание: фигура будет невидимой, пока не будет вызван //  
любой из этих трех методов.  
context.fill();  
context.stroke();  
context.closePath();
```

Функция `beginPath` позволяет нарисовать путь фигуры

Функция `lineTo` задает координаты для будущей фигуры

Canvas

- Метод `drawImage` позволяет вставлять другие изображения (элементы `img` и `canvas`) в контекст холста. В браузере Opera можно также рисовать изображения SVG внутри холста. Это достаточно сложный метод, который получает три, пять или девять аргументов:
- Три аргумента: Базовый вариант `drawImage` использует один аргумент для указания на включаемое изображение, и два для определения координат места назначения внутри контекста холста.
- Пять аргументов: Второй вариант использования `drawImage` включает приведенные выше три аргумента, плюс два для определения ширины и высоты вставляемого изображения (если вы захотите изменить его размер).
- Девять аргументов: Самый развитый вариант использования `drawImage` включает кроме пяти аргументов два значения для координат внутри исходного изображения, и два значения для ширины и высоты внутри исходного изображения. Эти значения позволяют динамически обрезать исходное изображение перед вставкой в контекст холста.

Canvas

```
// Три аргумента: элемент, координаты места назначения  
(x,y).  
context.drawImage(img_elem, dx, dy);  
// Пять аргументов: элемент, координаты места назначения  
(x,y)  
// и ширина и высота места назначения (если вы хотите  
изменить // размер исходного изображения).  
context.drawImage(img_elem, dx, dy, dw, dh);  
// Девять аргументов: координаты места назначения (x,y),  
// ширина и высота источника (для обрезки),  
// координаты места назначения (x,y),  
// ширина и высота места назначения (изменение размера).  
context.drawImage(img_elem, sx, sy, sw, sh, dx, dy, dw, dh);
```