Введение в HTML5

ст.преподаватель кафедры ПБИ
Еремеев Алексей Александрович

HTML5:

- Большей частью обратно совместим с тем, что там уже есть — не требуется учить совершенно новые языки для использования HTML5. Новые средства разметки работают таким же образом, как и старые (хотя семантика некоторых элементов изменилась – мы рассмотрим эти различия в будущей статье), и новые API основываются большей частью на том же JavaScript/DOM, который разработчики программировали в течение многих лет.

HTML5:

- Добавляет новые мощные средства в HTML, которые были ранее доступны в Web только с помощью технологии плагинов, такие как Flash, или с помощью сложного кода JavaScript и специальных приемов. Проверка форм и видео являются лучшими примерами.

- Лучше подходит для написания динамических приложений, чем предыдущие версии HTML (HTML был создан первоначально для создания статических документов).

HTML5:

- Имеет четко определенный алгоритм синтаксического анализа, так что все браузеры, реализующие HTML5, будут создавать одинаковое дерево DOM из одной и той же разметки, независимо от правильности. Это огромный выигрыш для совместимости.

Свойства HTML5

- **Новые семантические элементы**: *Семантика* является очень важной в *HTML* — мы всегда должны использовать для работы подходящий элемент. В *HTML* 4.01 мы имеем проблему — да, существует много элементов для определения специальных средств, таких как таблицы, списки, *заголовки*, и т.д., но существует также много общих свойств *web*-страницы, которые не имеют элемента для их определения.

Представление:

- HTML5 содержит новые семантические элементы, такие как <nav>, <header>, <footer> и <article>.

Свойства HTML5

- **Новые свойства форм**: *HTML* 4.01 уже позволяет создавать удобные, доступные *web*-формы, но некоторые общие свойства форм являются не слишком удобными и требуют специальных усилий для реализации. HTML5 предоставляет стандартизованный, простой способ реализации таких свойств, как выбор даты, ползунки и клиентская проверка.

Свойства HTML5

- **Собственная поддержка видео и аудио**: В течение многих лет видео и аудио в *Web* делалось, вообще говоря, с помощью *Flash*.
- HTML5 содержит элементы <video> и <audio> для простой реализации собственных видео и аудио плееров с помощью только открытых стандартов, и также содержит *API*, позволяющий легко реализовать индивидуальные *элементы управления* плеером.

Свойства HTML5

- **Рисования на холсте**: Элемент <canvas> позволяет определить на странице область для рисования, и использовать команды JavaScript для рисования линий, фигур и текста, импорта и манипуляций с изображениями и видео, экспорта в различные форматы изображений, и многих других вещей.

Свойства HTML5

- **Автономные приложения web**: HTML5 предоставляет ряд свойств, позволяющих приложениям *web* выполняться в автономном режиме. Кэши приложений позволяют сохранить копию всех ресурсов и других файлов, необходимых для локального выполнения приложения *web*. *Приложение* можно использовать, когда отсутствует соединение с сетью, и затем синхронизировать изменения с основной версией на сервере, когда *сеть* снова становится доступной.

Свойства HTML5

- **Геолокация**: Спецификация геолокации (не являющаяся частью спецификации HTML5) определяет *API*, который позволяет приложению *web* легко получить *доступ* к данным в любом местоположении, которое стало доступным, например, с помощью средств *GPS* устройства.

Введение в структурные элементы HTML5

- HTML4 имеет множество семантических элементов, позволяющих четко определять различные свойства страницы *Web*, такие как формы, списки, параграфы, таблицы и т.д. Однако он имеет и свои недостатки. Существенно используются элементы и с различными атрибутами id и class для определения различных других свойств, таких как навигационные *меню*, верхние и нижние колонтитулы, основной *контент*, *окна предупреждения*, боковые панели, и т.д.

Проблемы:

- Люди могут понять разницу между различным контентом, но машины не могут — браузер не видит различные div как верхние и нижние колонтитулы и т. д. Он видит их как различные div.
- Даже если для решения некоторых из этих проблем используется дополнительный код, вы можете сделать это надежно только для собственных web-сайтов, так как другие разработчики web будут использовать другие имена классов и ID, особенно, если рассмотреть международную аудиторию — различные разработчики web в разных странах будут использовать различные языки для записи имен своих классов и id.

Новые тэги

- <header>
- <*footer*>
- <*nav*>
- <article>
- <section>
- <time>
- <aside>
- <hgroup>
- <*figure*> и <figcaption>

Мета-различия

- **HTML 4.01:**
- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

- **HTML5:**
- <!DOCTYPE html>

<header>

- *Верхний колонтитул* документа выглядит следующим образом:

```
<header>
    <hgroup>
        <h1>A history of Pop Will Eat Itself</h1>
        <h2>Introducing the legendary Grebo Gurus!</h2>
    </hgroup>
</header>
```

- Назначение элемента <header> состоит в создании оболочки вокруг раздела контента, который формирует *верхний колонтитул* страницы, содержащий обычно логотип компании/графику, заголовок основной страницы, и т.д.
- <hgroup> - позволяет учитывать группу заголовков как один заголовок в структуре документа.

<footer>

```
<footer>
   <h3 id="copyright">Copyright and attribution</h3>
</footer>
```

<*footer*> необходимо использовать для размещения контента нижнего колонтитула сайта.

<nav>

- Элемент <*nav*> предназначен для разметки навигационных ссылок или других конструкций (например, формы поиска), которые направляют вас на различные страницы текущего сайта, или различные области текущей страницы. Другие ссылки, такие как рекламные ссылки, не учитываются.

<aside>

- Элемент <aside> предназначен для разметки фрагментов контента, которые имеют отношение к основному контенту, но не вписываются явно в основной поток изложения. Например, в данном случае мы имеем пакет кратких интересных фактов и статистики о музыкальной группе, которые не очень хорошо подходят для основного контента. Другими подходящими кандидатами для элементов <aside> являются списки ссылок на внешний связанный контент, справочная информация, цитаты, и боковые панели.

<figure> и <figcaption>

- Элемент <*figure*> хорошо подходит, чтобы объединить весь контент, из которого вы хотите составить один рисунок, будет ли это текст, изображения, SVG, видео, или что-то другое. Элемент <figcaption> затем помещается внутри элемента <*figure*>, и содержит описательный заголовок для этого рисунка.

<time>

- Элемент <time> позволяет определить точно выраженное значение даты и времени, которое одновременно понятно человеку и машине.
- Текст между открывающим и закрывающим тегами может быть любым, подходящим для людей, посещающих сайт. При желании можно сделать это следующим образом:

1. <time datetime="1989-03-13">13th March 1989</time>
2. <time datetime="1989-03-13">March 13 1989</time>
3. <time datetime="1989-03-13">My nineteenth birthday</time>

<article> и <section>

- В основном элемент <article> предназначен для независимых фрагментов контента, которые будут иметь смысл вне контекста текущей страницы, и могут хорошо объединяться. Такие фрагменты контента включают публикации в блоге, видео и его текстовая запись, новостная история, или одна часть серийной истории.
- Элемент <section>, с другой стороны, предназначен для разбиения контента страницы на различные функциональные или тематические области, или разбиения статьи или истории на различные части.

Как заставить это работать в старых браузерах

Для всех браузеров, кроме одного, достаточно прописать в CSS:

```
article, section, aside, hgroup, nav, header,
footer, figure, figcaption
 { display: block; }
```

Кроме IE

```
<script>
   document.createElement('article');
document.createElement('section');
document.createElement('aside');
document.createElement('hgroup');
document.createElement('nav');
document.createElement('header');
document.createElement('footer');
document.createElement('figure');
document.createElement('figcaption'); </script>
```