**Введение**

HTML5 содержит две новые вещи: новые API, которые добавляют существенно важные новые свойства к открытым стандартам модели разработки web, и новые структурные элементы, которые определяют специальные свойства страницы Web со значительно более точной семантикой, чем это было доступно в HTML 4. Статьи, посвященные многим новым API, можно найти на Dev.Opera (<http://dev.opera.com/articles/tags/html5/>) с меткой HTML5.

Данная статья рассматривает другую составляющую – мы кратко опишем, как были выбраны новые семантические элементы, какие новые свойства имеются, и как они используются, как в HTML5 работают заголовки, и поддержку в браузерах новых элементов, включая их поддержку в старых браузерах.

**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 <div> и <span> 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. Что-нибудь типа <div id="header"> будет понятно разработчикам и дизайнерам, знающим, для чего это предназначено, и умеющих использовать CSS и JavaScript для применения собственных стилей и поведения, чтобы сделать это понятным для конечных пользователей.

Все могло бы быть значительно лучше, но с такими настройками по-прежнему существуют 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. Не будет ли полезнее, если браузеры и считыватели экрана смогут явно идентифицировать, скажем, навигационное меню, чтобы пользователям с недостатками зрения было легче его найти, или различные элементы новостей в пакете блогов, чтобы их было легче объединить в ленту RSS без дополнительного программирования?
* 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.

Поэтому имеет смысл определить согласованное множество элементов для общих структурных блоков, которые часто появляются на большинстве Web-сайтов. В данной статье будут рассмотрены следующие новые элементы HTML5:

* <header>: Используется для верхнего колонтитула сайта.
* <footer>: Используется для нижнего колонтитула сайта.
* <nav>: Содержит навигационные функции страницы.
* <article>: Содержит автономный фрагмент контента, который будет иметь смысл, если используется как позиция RSS, например, новостное сообщение.
* <section>: Используется либо для объединения в группу различных статей с различной целью или по различным темам, или для определения различных разделов одной статьи.
* <time>: Используется для разметки времени и даты.
* <aside>: Определяет блок контента, который связан с основным контентом, но не входит в его основной поток.
* <hgroup>: Используется в качестве оболочки скрытия более одного заголовка, если требуется, чтобы учитывался только один заголовок в структуре заголовков страницы.
* <figure> и <figcaption>: Используется для инкапсуляции рисунка как единого элемента, и содержит, соответственно, подпись для рисунка.

**<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>**

Вы можете заметить, что в приведенном выше коде, единственным контентом колонтитула является элемент <hgroup>, содержащий два заголовка. Я хочу здесь определить заголовок документа верхнего уровня, плюс подзаголовок/ключевую фразу. Мне нужно, чтобы только заголовок верхнего уровня учитывался в иерархии заголовков документа, и именно это делает элемент <hgroup> - он позволяет учитывать группу заголовков как один заголовок в структуре документа. Подробнее о работе иерархии заголовков в HTML5 будет написано ниже в разделе "Разбивка HTML5 и алгоритм заголовков HTML5".

**<footer>**

Внизу документа можно увидеть следующий код:

<footer>

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

</footer>

<footer> необходимо использовать для размещения контента нижнего колонтитула сайта — если просмотреть нижнюю часть какого-то количества сайтов, то легко увидеть, что нижние колонтитулы могут содержать различные вещи, от уведомления об авторских правах и контактной информации, до заявления о доступности, информации о лицензировании и множества других второстепенных ссылок.

**Примечание**: Не существует ограничения только на один верхний и нижний колонтитул на страницу — страница может содержать несколько статей, и иметь для каждой статьи верхний и нижний колонтитул.

**<nav>**

Двигаясь дальше по документу, мы встречаемся со следующей структурой:

<nav>

<h2>Contents</h2>

<ul>

<li><a href="#Intro">Introduction</a></li>

<li><a href="#History">History</a>

<!—другие навигационные ссылки ... -->

</ul>

</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. Можно, конечно, включать заголовки и другие структурные элементы внутрь <nav>, но это не обязательно.

**<aside>**

Сразу под заголовком документа имеется следующий код:

<aside>

<table>

<!—- множество кратких фактов в этом месте -->

</table>

</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>**

Динамический дуэт из <figure> и <figcaption> был создан для решения достаточно специального множества проблем. Прежде всего, не казалось ли всегда немного семантически сомнительно и неясно размечать рисунок и его подпись как два параграфа, или как пару списка определений, или как-то иначе? И второе, что делать, когда требуется рисунок, состоящий из изображения, или двух изображений, или двух изображений и некоторого текста? 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. В мой пример включен простой рисунок, чтобы просто показать, как это используется:

<figure>

<img src="pwei.png" alt="Old poppies logo" />

<figcaption>

The old poppies logo, circa 1987.<br /> <a href="http://www.flickr.com/photos/bobcatrock/317261648/">

Original picture on Flickr</a>, taken by bobcatrock.

</figcaption>

</figure>

#### <time>

Элемент <time> позволяет определить точно выраженное значение даты и времени, которое одновременно понятно человеку и машине. Например, я пометил даты выпуска синглов музыкальной группы следующим образом:

<time datetime="1989-03-13">1989</time>

Текст между открывающим и закрывающим тегами может быть любым, подходящим для людей, посещающих сайт. При желании можно сделать это следующим образом:

<time datetime="1989-03-13">13th March 1989</time>

<time datetime="1989-03-13">March 13 1989</time>

<time datetime="1989-03-13">My nineteenth birthday</time>

С другой стороны дата внутри атрибута datetime представлена в стандарте ISO и является машинно-читаемой датой, поэтому мы получаем двойную выгоду. Можно также добавить время в конце стандартного представления ISO следующим образом:

<time datetime="1989-03-13T13:00">One o'clock in the afternoon, on the 13th of March 1989</time>

Можно добавить также настройку часового пояса, поэтому, например, чтобы представить последний пример в стандартном тихоокеанском времени, можно сделать следующее:

<time datetime="1989-03-13T13:00Z-08:00">One o'clock in the afternoon, on the 13th of March 1989</time>

#### <article> и <section>

Теперь обратим наше внимание к двум, возможно, наиболее неправильно понимаемым элементам HTML5 - <article> и <section>. При первом знакомстве различие может показаться нечетким, но на самом деле не все так плохо.

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. Поэтому, например, в моей истории PWEI, структура выглядит следующим образом:

<article>

<section id="Intro">

<h2>Introduction</h2>

</section>

<section id="History">

<h2>History</h2>

</section>

<section id="Discography">

<h2>Discography</h2>

</section>

</article>

Но можно также структурировать следующим образом:

<section id="rock">

<h2>Rock bands</h2>

<!--multiple article elements could go in here -->

</section>

<section id="jazz">

<h2>Jazz bands</h2>

<!--multiple article elements could go in here -->

</section>

<section id="hip-hop">

<h2>Hip hop bands</h2>

<!--multiple article elements could go in here -->

</section>

##### Где остается <div>?

Итак, со всеми этими прекрасными новыми элементами, которые можно использовать на страницах Web, дни простого элемента <div> сочтены, не так ли? НЕТ. Фактически, <div> по-прежнему имеет совершенно законное применение. Его следует использовать, когда не существует другого более подходящего доступного элемента для объединения области контента, что часто происходит, когда вы используете элемент только для объединения контента в группу с целью стилевого или визуального оформления. Примером в моей истории PWEI является элемент <div id="wrapper">, который использован для создания оболочки вокруг всего контента. Единственная причина его использования здесь в том, чтобы я мог использовать CSS для выравнивания контента по центру в браузере:

#wrapper {

background-color: #ffffff;

width: 800px;

margin: 0 auto;

}

##### <mark>

Элемент <mark> предназначен для выделения терминов, значимых в данный момент, или выделения частей контента, к которым вы просто хотите привлечь внимание, но не хотите изменять семантическое значение. Это похоже на то, как при просмотре напечатанной статьи вы выделяете важные для вас строки с помощью цветного маркера. Поэтому, например, вы можете захотеть использовать этот элемент для выделения строк в wiki, которые требуют редакторской правки, или выделения экземпляров термина поиска, который пользователь только что искал на странице, и задание затем для них подходящего оформления в CSS.

##### Атрибут hidden

Атрибут hidden, когда применяется к любому элементу, скрывает его полностью от любых форм представления/медиа, и должен использоваться, если вы намерены показать контент позже (например, используя JavaScript для удаления этого атрибута), но не хотите, чтобы он отображался в данный момент. Он не должен использоваться для скрытия такого контента, как скрытые вкладки интерфейса с вкладками, так как это совершенно другой способ представления контента в меньшем пространстве, а не скрытие контента вообще.

### Разбивка HTML5 и алгоритм заголовков HTML5

Прежде чем продолжить путешествие в направлении более глубокого овладения HTML5, необходимо обсудить одно важное отличие, которое существует между HTML5 и предыдущими версиями спецификации. В HTML мы имеем концепцию разбивки документа, которая является в основном разбивкой документа по его заголовкам, и их иерархии относительно друг друга, точно таким же образом, как при записи документа в текстовом процессоре вы просматриваете документ в виде плана-конспекта. На самом деле я фактически создал план-конспект этого документа, вкладывая списки друг в друга для создания оглавления в начале статьи. План-конспект этой статьи выглядит примерно следующим образом:

* Введение в структурные элементы HTML5
  + Как были выбраны названия элементов?
  + Почему нет элемента <content>?
* Представляем пример страницы HTML5
  + Некоторые мета-различия
  + <header>
  + <hgroup>
  + <footer>
  + <nav>
  + <aside>
  + <figure> и <figcaption>
  + <time>
  + <article> и <section>
    - Где остается <div>?
    - <mark>
    - Атрибут hidden
* Разбивка HTML5 и алгоритм заголовков HTML5
* Как заставить это работать в старых браузерах
* Заключение

Поэтому "Новые структурные элементы в HTML5" будет <h1>, "Введение в структурные элементы HTML5" будет <h2>, и т.д. В HTML4 мы привыкли к тому факту, что имеется шесть возможных уровней заголовков, и каждый уровень заголовков диктуется реально используемым элементом, что означает, что вполне возможно получить совершенно искаженную иерархию заголовков, если использовать неправильные уровни заголовков, или даже если часть вашего контента объединяется в другой CMS (Системе управления контентом).

HTML5 решает эту проблему, создавая иерархию заголовков на основе относительного вложения различных разделов документа. Новый раздел документа создается, когда вы используете так называемый **разделяющий контент** – элементы <article>, <section>, <nav>, и <aside>. Возьмем следующий пример.

<h1>My title</h1>

<div>

<h2>My subtitle</h2>

</div>