Annals of DAAAM for 2012 & Proceedings of the 23rd International DAAAM Symposium, Volume 23, No.1, ISSN 2304-1382
ISBN 978-3-901509-91-9, CDROM version, Ed. B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2012
Make Harmony between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals & Proceedings of DAAAM International 2012

STRUCTURING DOCUMENTS WITH NEW HTML5 SEMANTIC ELEMENTS

FULANOVIC, B[ojan]; KUCAK, D[anijel] & DJAMBIC, G[oran]

Abstract: *This paper will explore howto use new semantic tags in HTML5 (Hyper Text Markup Language) and how to structure document to create meaningfulhtml document outline.It will explore new semantic elements and cover changes in syntax of previous versions of HTML. Also it will describe HTML5 outline algorithm that helps defining and structuring site content. It willstudy statistics of naming html tags justifying the increasing use of HTML5 semantic tags as a standard in the structuring ofhtmldocuments.*
Keywords:*html5, structure, semantics, syntax, web site*

1. INTRODUCTION

The first standardized version of HTML [1] (Hyper Text Markup Language)occurred in 1995 with release of HTML 2.0 HTML 2.0 has standardized the rules that web community was already using.In 1997 the version of HTML 3.2 was released. It didn't bring anything spectacular but only a support for specific tags such as tables which browsers were already supporting. At that time all browser manufacturers werereleasing software updates that supported their own tags and offers a separate level of support for certain existing features. All this has resulted in a tremendous lot of work and adjustment for designers and developers. Typically multiple versions of web sites had to be developed to support as much internet browsers as possible.In the year 1998 a group of web developers started the Web Standards Project[2], which dedicated for the greater standardization in developing as recommended by the W3C (World Wide Web Consortium)[3] consortium, moreover, they have promoted all internet browsers that adopted those standards. All this resulted in the release version of HTML 4.0which stabilized the language and became a standard until today. In 2000 XHTML 1.0 (Extensible HyperText Markup Language) was introduced.The W3C consortium tried to further structure HTML in a way to introduce XML (Extensible Markup Language)[4] rules.Designers and developers did not adopt XHTML as expected because it isn't backwards compatible. Also all errorsfound on the pagerelating to improper formatting are defined as fatal which causes user agent to stop processing the web page and display the error message to the user. In 2009 W3C gave up from XHTML 2.0 Group of developers including Mozilla and Opera software foundation who did not like the direction in which XHTML was going founded the WHATWG (Web Hypertext Application Technology Working Group) [5] which continued the development of HTML. In 2007 W3C adoptedthe WHATWG work under the name HTML5 [6]. HTML5 reflects needs of a modern web design. With HTML4 most of functionality is achieved with JavaScript [7] and third party plugins.

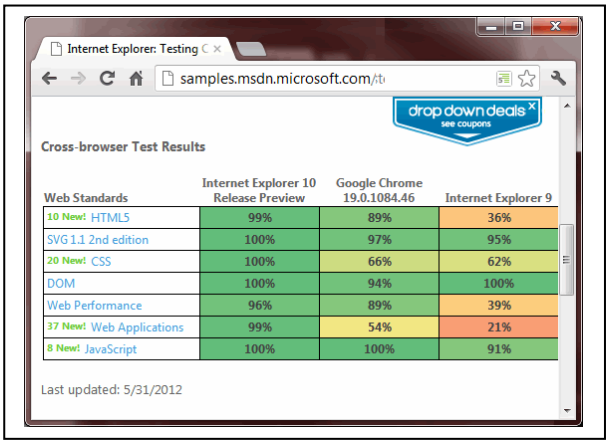


Fig. 1. Cross-browser HTML5 support

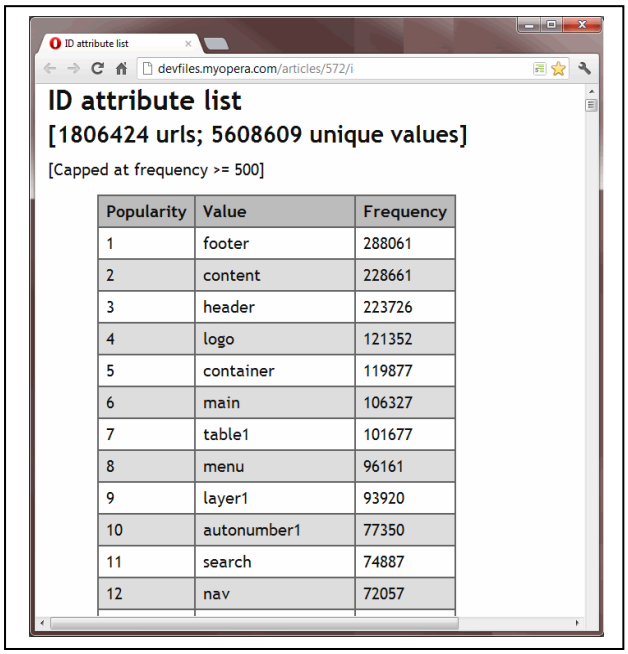
HTML 5 brings much of that functionality directly to browsers. Most responsible for the rapid breakthrough of HTML5 today are companies like Apple, Microsoft and Google which added support for HTML5 in internet browsers and mobile devices.Although HTML5 as a standard has not yet been accepted by the W3C consortium, all major browser has implemented many of HTML 5 features Fig.1. It means that HTML5 time has come and it is very important how future html web projects will be structured.

2. HTML5 SYNTAX

Web developers who write the code for XHTML had to follow much of coding rules. Some of rules are that all tag names are in lowercase, tags that use attributes should have attribute values inside quotation marks, tags that don't have any content are self-closed tags (they don't have closing tag). In HTML5 rules have drastically changed and it's very loose compared to XHTML. For example, opening tag can be written in capital letters and closing tag in lower, attribute values can or cannot be inside quotation marks, attributes with *boolean*values can stand alone without attribute value if a value is true. You can also capitalize a letter inside tag name and leave other in lower case. *Html*, *head* and *body* tags are optional and tags like paragraphs, list items, table rows don’t have to be necessary closed. All pages written by these rules normally pass validation. In HTML5 there are many options how we can write our code. Unfortunately the downside of this flexibilityis that many developers write ugly, hard to read code. Regardless of the flexibility,a code should be written in a way to be consistent and readable not just for author of html

```
<!DOCTYPE html>
<html>
<head>
    <title>My HTML5 photo site</title>
</head>
<body>
<div id="header">
    <h1>Welcome to my photo site</h1>
</div>
<div id="navigation">
    <ul>
        <li><a href="News.aspx">News</a></li>
        <li><a href="Cameras.aspx">Cameras</a></li>
        <li><a href="Reviews.aspx">Reviews</a></li>
    </ul>
</div>
<div id="mainContent">
    <h2>Canon EOS 5D Mark III</h2>
    <p>The latest incarnation of Canon's enthusiast full-frame 5D
</div>
<div id="sidebar">
    <h2>Photo equipment</h2>
    <h3>Camera tripod</h3>
    <a href="Order.aspx">Order...</a>
</div>
<div class="footer">
    <p>My photo site &copy;2012</p>
</div>
</body>
</html>
```

Fig. 2 Naming tags with id and class attributes



Popularity	Value	Frequency
1	footer	288061
2	content	228661
3	header	223726
4	logo	121352
5	container	119877
6	main	106327
7	table1	101677
8	menu	96161
9	layer1	93920
10	autonumber1	77350
11	search	74887
12	nav	72057

Fig. 3 ID attribute list

document but to other developers too. For example in all html documents included in a project developer should maintain the same rules like keep lower case tags, keep attributes values inside quotation marks or not includingattributes values for thosewho have *boolean* value. Keeping these rules in mind the code will be easy to read for all developers in web community and most important it's backwards compatible.

3. PURPOSE OF SEMANTIC ELEMENTS

Standard tag for grouping elements in HTML4 is the *div* tag that represents a generic block element and it has no semantic meaning at all. Its purpose is to enable us to group elements. To add meaning to *div* elements, web developers are using the *id* and *class* attributes (each tag can contain several attributes that give additional meaning to particular tag) Fig.2.The problem with the names we assign to *id* and *class* attributes is that there is no standard in naming procedure. There is no specification for naming attributes on which basis some user agent would be able to understand html content they are using. Various companies have started to elaborate studies on names that have been used for assigning *id* and *class* attributes Fig. 3.Looking at the results of these studies there could be easily observed most frequently used patterns such as *footer*, *header*, *nav* etc. These names were the main encouragement for creating new semantics tags in HTML5.For example, on Fig. 4, even if we don't know anything about tags it's very easy to find where is the navigation on the page, we just need to find a *nav* tag. Also if we use a number of identical semantic tags, it is absolutely legal to use the *id* or *class* attributes for further clarification of the element purpose.It should be emphasized that the use of semantic tags does not mean anything to viewers. Browsers and accessibility devices like screen readers will notice a difference and that is the point of semantic markup. If we compare HTML4 code Fig.2 with HTML5 code Fig. 4 which use semantic tags, it is clear to see how much easier is to manage HTML5 code.

4. NEW SEMANTIC ELEMENTS

Due to many options that HTML5 gives us knowing which element to use in particular situation is in most cases a matter of personal judgment. For developer it's very important to learn from HTML5 specification [8] everything about new semantic elements to be aware of all options available and to make right judgment call.

3.1 Header element (<header>)

The header element is used for representing a group of introductory and navigational aids. Most common use of a*header* element is for putting headings inside (*h1-h6* and *hgroup* element). Also it can contain navigation, sections table of content, search form or company logo. Inside the HTML4 code in Fig. 2 we replaced generic *div* element with the*id* value of *header* with *header* semantic element as it's shown in Fig. 4

3.2 Nav element (<nav>)

As the name suggest the purpose of *nav* element is to identify site navigation that links to other pages or sections within the page. If there are more parts on the page that contain links, not all of the parts should group links within the *nav* element. Using the *nav* element is recommended for major navigation links. Users with poor eyesight which use user agents like screen readers will have large benefit of using *nav* element for major navigation. It will help them to determine which content on the page they are interested in. The *Nav* element doesn't have to contain only the navigation links, many other elements like heading, paragraph etc are allowed. Inside the HTML4 code in Fig. 2 we replaced a generic *div* element with the *id* value of *navigation* with *nav* semantic element as it is shown in Fig. 4

```
<!DOCTYPE html>
<html>
<head>
    <title>My HTML5 photo site</title>
</head>
<body>
<header>
    <h1>Welcome to my photo site</h1>
</header>
<nav>
    <ul>
        <li><a href="News.aspx">News</a></li>
        <li><a href="Cameras.aspx">Cameras</a></li>
        <li><a href="Reviews.aspx">Reviews</a></li>
    </ul>
</nav>
<article>
    <h2>Canon EOS 5D Mark III</h2>
    <p>The latest incarnation of Canon's enthusiast full-frame 5D
</article>
<aside>
    <h2>Photo equipment</h2>
    <h3>Camera tripod</h3>
    <a href="Order.aspx">Order...</a>
</aside>
<footer>
    <p>My photo site &copy;2012</p>
</footer>
</body>
</html>
```

Fig. 4 Using semantic markup

3.3 Section element (<section>)

It represents the section of a document which groups thematically related content. It doesn't represent a generic container element which the author can use for styling purposes, in such a case the *div* element should be used. If authors won't use a *section* content on other pages and if a *section* content will be listed in the document's outline it is appropriate to use it.

3.4 Article element (<article>)

The article element represents a independent item of the content. There is a lot of confusion when to use the section and when to use the article element. The best way to distinguish these two elements is to analyze its content. If it's content that can be republished or syndicated or its content that user agent can access and can understand without the rest of the page than the *article* element is appropriate. Most examples using *article* elements are blog posts, newspaper articles, user submitted content etc. If *article* elements are nested they have to be related to the parent article element.

3.5 Aside element (<aside>)

Represents a section of a page with content related to the other content on the page, Also it could be considered separated from that content. A common use is to place advertising content or group *nav* elements. Inside the HTML4 code in Fig. 2 we replaced generic *div* element with *id* value of *sidebar* with *aside* semantic element as it is shown in Fig. 4

3.6 Footer element (<footer>)

The *Footer* element represents a footer for its sectioning a root element. It means if you find a *footer* inside the *body* element then it is a footer for the entire page, if we found the *footer* inside the *article* element that it's footer for that element. The common opinion that a footer should be at the very end of the element is wrong, it can be anywhere. Pages and sections can have multiple footers. Also, the*footer* can contain a complex element structure inside.

3.7 Deciding which element to use

In most cases it is obvious which element should be used in particular situations but in case of ambiguous content choosing right element could be tricky. A good way is to ask yourself a question about the content. Answer can lead you to choose the right one. Major point is to decide if a content represents a new section. If not, check to see if a the content contains related links or contact information about web author or has any legal information. In this case using *footer* element is the right choice. If the content represents the introductory part you should use the *header* element. In the case when there is a grouped content for stylistic reasons or it is grouped to extend its meaning trough *id* or *class* attribute the *div* element is appropriate. If the content requires a new section developer should consider next few questions. Is it thematically grouped content? Choose the *section* element. Is the content related to the other content on the page? Choose the *aside* element. Does the content contain links? If the answer is yes than check whether those links are major links on the page? If it is the case, use the *nav* element. Is the content self-contained or can it be presented and understood on other pages by its own? Choose the *article* element.

5. DOCUMENT STRUCTURE IN HTML5

HTML5 has many algorithms that instruct user agents how to handle html documents. Algorithms are handling parsing code, error handling, and parsing badly written code. Outline algorithm handles how sectioning content should be parsed to build the outline of the current document. By using the rules for outlining thealgorithm we can be sure that our page is structured exactly the way we want it to be.This can be useful for accessibility reasons or to make the content easy to distribute.To understand the outline algorithm, the best way is to imagine web page as a table of content where important sections are displayed as individual items and list interior sections nested within them Fig. 5.

1. News
    a. Nokia Lumia 920 promises PureView
    b. Corel announces PaintShop Pro X5
    c. Carl Zeiss presents CZ.2 28-80mm
2. Cameras
    a. Canon
        i. 400D
        ii. 5D Mark 2
    b. Nikon
        i. D3000
        ii. D300S
3. Reviews
    a. Canon EOS Rebel T4i
    b. Samsung NX210

Fig. 5 Document outline

```
<doctype html>
<title>My photo site</title>
<h1>News</h1>
<h1>Cameras</h1>
<h1>Reviews</h1>
```

1. News
2. Cameras
3. Reviews

Fig. 6 Outlining the document with headings

HTML5 examine how section and heading content are used to define the outline. Algorithm starts with the *body* element and established him as the outline root. After that parser goes through other elements in a document to establish its structure. When new section is found new item is added to the outline. Sections containing the heading content are named with that heading. Because sections are containers any new section created within the existing one are nested in the outline. Elements that create new sections are *section*, *nav*, *article*, *aside* and heading elements. First heading element used within a section element is used for defining heading for that section. Any additional headings will create new nested headings based on their rank. Outlining document is important for search engines, assistive technologies, screen readers etc.Poorly formed documents have a confusing table of content, search engines won't be able to properly rank page data and navigation for assistive technologies is difficult. The rules governing strategies concerning headings use have changed from HTML4 to HTML5. In HTML4 the only way to section page was to use different heading rankings and knowing how to use headings was important because it defines a document structure. Also authors were using only one *h1* heading that will title the page and assist search engine rankings.In HTML5 rules are a bit different. Individual sections have their own distinct hierarchy apart from the page which means it is absolutely legal to use more than one *h1* heading element per page.Although the HTML5 specification recommends using more than one *h1* per page most developers are going against specifications because search engines still strongly value how many *h1* there are on the page. Until search engines adopt these rulesmaybe using *h2* on nested sections will be a good practice. Headings are creating implicit sections. As Fig.6 shows if we put three *h1* heading one after another all three will create a new section and they will be on the same level because of the same ranking. First *h1* will be the title of the body element because the first heading used inside the section represents the title of the section. If we put *h2* heading inside section elements it will result in nested sections on page outline Fig. 7. If section elements are removed we can achieve the same outline structure by changing heading rankings Fig. 7. There is no exact answer which way is better.How the document structure will be outlined depends on the author's preferences.It is good to create a policy that will follow the size, structure and complexity of the web site. It is always desirable, no matter which style you choose to be consistent trough the code inside your web project.To ensure and proof that the web page is outlined correctly developers can use several online applications. One of them is the *HTML5 Outliner* [9] that goes through your page code and parses it using HTML5 outline algorithm.You can paste your code directly into online form, pass URL (Uniform Resource Locator) to the html document or upload the html document to the server.Another way to test outlining is to install extensions to your browser or

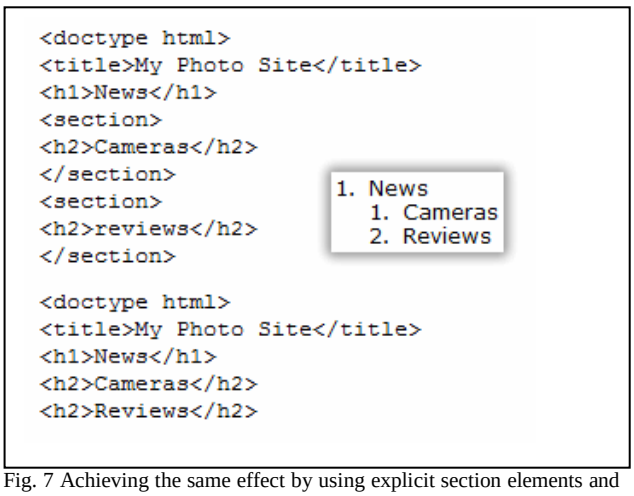


Fig. 7 Achieving the same effect by using explicit section elements and heading ranking

even to download existing minified JavaScript file and build your own parser.

6. CONCLUSION

Right now it is very interesting time in development of the web. Mobile devices suppress desktop PCs in terms of Internet browsing. Evolution of HTML5 is not revolutionary but an upgrade of existing technology. If everything was as it was meant to be the HTML5 would facilitate the development of web applications, but in practice, things are a little different. Currently on the scene we have mobile devices that are fighting for supremacy, big market players such as Apple and Google, which are constantly changing standards in order to enhance their products and dominate the market. The same thing happens in browser marketplace.

Based on mentioned above it can be concluded that the development of the web didn't go far away from the beginning. Web development was always disordered and changes in web technologies are constant. HTML5 promises that things will be somewhat standardized. Things indicating this fact are very rapid adoption of HTML5 specifications in browsers on desktop and mobile devices..

7. REFERENCES

[1] Musciano C.; Kennedy B. (2010), *HTML & XHTML: The Definitive Guide*,O'Reilly Media, Inc., 978-0-596-52732-7

[2] http://www.webstandards.org/ , The Web Standards Project, *Accessed: 08-11-2012*

[3] http://www.w3.org/ , The world wide Web Consortium, *Accessed: 09-02-2012*

[4] Newcomer, E. (2003), *Understanding Web Services: XML, WSDL, SOAP and UDDI*,Independent Technology Guides

[5] http://www.whatwg.org/ , WHATWG, *Accessed: 06-27-2012*

[6] Pilgrim, M. (2010).*HTML5 Up and running*,O'Reilly Media, Inc., 978-0-596-80602-6

[7] Flanagan, D. (2004), *JavaScript: The Definitive Guide, 4th edition*,O'Reilly Media, Inc., Sebastopol, California 95472

[8] http://dev.w3.org/html5/spec/single-page.html , HTML 5 Specification, *Accessed: 09-02-2012*

[9] http://gsnedders.html5.org/outliner/, HTML 5 Outliner, *Accessed: 07-07-2012*