Sections and Outlines of an HTML5 Document

> ***Note**: There are currently no known implementations of the outline algorithm in graphical browsers or assistive technology user agents, although the algorithm is implemented in other software such as conformance checkers. Therefore the outline algorithm cannot be relied upon to convey document structure to users. Authors are advised to use heading rank (h1-h6) to convey document structure.*

The HTML5 specification brings several new elements to web developers allowing them to describe the structure of a web document with standard semantics. This document describes these elements and how to use them to define the desired outline for any document.

Structure of a Document in HTML 4

The structure of a document, i.e., the semantic structure of what is between <body> and </body>, is fundamental to presenting the page to the user. HTML4 uses the notion of sections and sub-sections of a document to describe its structure. A section is defined by an HTML Dividing ( ) element with HTML Heading Elements (<h1>, <h2>, <h3>, <h4>, <h5>, or <h6>) within it, defining its title. The relationships of these HTML Dividing and HTML Heading Elements leads to the structure of the document and its outline.

So the following mark-up:

```
<div class="section" id="forest-elephants" >
  <h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser known forest elephants.
    ...this section continues...
  <div class="subsection" id="forest-habitat" >
    <h2>Habitat</h2>
    <p>Forest elephants do not live in trees but among them.
     ...this subsection continues...
  </div>
</div>
```

leads to the following outline:

```
1. Forest elephants
   1.1 Habitat
```

The `<div>` elements aren't mandatory to define a new section. The mere presence of an HTML Heading Element is enough to imply a new section. Therefore,

```
<h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser known forest elephants.
    ...this section continues...
  <h2>Habitat</h2>
  <p>Forest elephants do not live in trees but among them.
    ...this subsection continues...
  <h2>Diet</h2>
<h1>Mongolian gerbils</h1>
```

leads to the following outline:

```
1. Forest elephants
   1.1 Habitat
   1.2 Diet
2. Mongolian gerbils
```

Problems Solved by HTML5

The HTML 4 definition of the structure of a document and its implied outlining algorithm is very rough and leads to numerous problems:

1. Usage of `<div>` for defining semantic sections, without defining specific values for the **class** attributes makes the automation of the outlining algorithm impossible ("Is that `<div>` part of the outline of the page, defining a section or a subsection?" Or "is it only a presentational `<div>`, only used for styling?"). In other terms, the HTML4 spec is very imprecise on what is a section and how its scope is defined. Automatic generation of outlines is important, especially for assistive technology, that are likely to adapt the way they present information to the users according to the structure of the document. HTML5 removes the need for `<div>` elements from the outlining algorithm by introducing a new element, `<section>`, the HTML Section Element.
2. Merging several documents is hard: inclusion of a sub-document in a main document means changing the level of the HTML Headings Element so that the outline is kept. This is solved in HTML5

as the newly introduced sectioning elements (`<article>`, `<section>`, `<nav>` and `<aside>`) are always subsections of their nearest ancestor section, regardless of what sections are created by internal headings.

3. In HTML4, every section is part of the document outline. But documents are often not that linear. A document can have special sections containing information that is not part of, though it is related to, the main flow, like an advertisement block or an explanation box. HTML5 introduces the `<aside>` element allowing such sections to not be part of the main outline.
4. Again, in HTML4, because every section is part of the document outline, there is no way to have section containing information related not to the document but to the whole site, like logos, menus, table of contents, or copyright information and legal notices. For that purpose, HTML5 introduces three specific sections elements: `<nav>` for collections of links, such as a table of contents, `<footer>` and `<header>` for site-related information.

More generally HTML5 brings precision to the sectioning and heading features, allowing document outlines to be predictable and used by the browser to improve the user experience.

The HTML5 Outline Algorithm

Defining Sections in HTML5

All content lying inside the `<body>` element is part of a section. Sections in HTML5 can be nested. Beside the main section, defined by the `<body>` element, section limits are defined either explicitly or implicitly. Explicitly-defined sections are the content within `<body>`, `<section>`, `<article>`, `<aside>`, `<footer>`, `<header>`, and `<nav>` tags.

> ***Note:*** *Each section can have its own heading hierarchy. Therefore, even a nested section can have an `<h1>`. See Defining Headings in HTML5.*

Example:

```
<section>
  <h1>Forest elephants</h1>
  <section>
    <h1>Introduction</h1>
    <p>In this section, we discuss the lesser known forest elephants.</p>
  </section>
  <section>
    <h1>Habitat</h1>
    <p>Forest elephants do not live in trees but among them.</p>
  </section>
  <aside>
```

```
    <p>advertising block</p>
  </aside>
</section>
<footer>
  <p>(c) 2010 The Example company</p>
</footer>
```

This HTML snippet defines two top-level sections:

```
<section>
  <h1>Forest elephants</h1>
  <section>
    <h1>Introduction</h1>
    <p>In this section, we discuss the lesser known forest elephants.</p>
  </section>
  <section>
    <h1>Habitat</h1>
    <p>Forest elephants do not live in trees but among them.</p>
  </section>
  <aside>
    <p>advertising block</p>
  </aside>
</section>

<footer>
  <p>(c) 2010 The Example company</p>
</footer>
```

The first section has three subsections:

```
<section>
  <h1>Forest elephants</h1>

  <section>
    <h1>Introduction</h1>
    <p>In this section, we discuss the lesser known forest elephants.</p>
  </section>

  <section>
    <h1>Habitat</h1>
    <p>Forest elephants do not live in trees but among them.</p>
  </section>

  <aside>
```

```
    <p>advertising block</p>
  </aside>
</section>

<footer>
  <p>(c) 2010 The Example company</p>
</footer>
```

This leads to the following structure:

```
1. Forest elephants
   1.1 Introduction
   1.2 Habitat
   1.3 Section (aside)
```

Defining Headings in HTML5

While the HTML Sectioning elements define the structure of the document, an outline also needs headings to be useful. The basic rule is simple: the first HTML heading element (one of `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`) defines the heading of the current section.

The heading elements have a *rank* given by the number in the element name, where `<h1>` has the *highest* rank, and `<h6>` has the *lowest* rank. Relative ranking matters only within a section; the structure of the sections determines the outline, not the heading rank of the sections. For example, this code:

```
<section>
  <h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser known forest elephants.
    ...this section continues...
  <section>
    <h2>Habitat</h2>
    <p>Forest elephants do not live in trees but among them.
        ...this subsection continues...
  </section>
</section>
<section>
  <h3>Mongolian gerbils</h3>
  <p>In this section, we discuss the famous mongolian gerbils.
     ...this section continues...
</section>
```

leads to the following outline:

```
1. Forest elephants
   1.1 Habitat
2. Mongolian gerbils
```

Note that the rank of the heading element (in the example `<h1>` for the first top-level section, `<h2>` for the subsection and `<h3>` for the second top-level section) is not important. (Any rank can be used as the heading of an explicitly-defined section, although this practice is not recommended.)

Implicit Sectioning

Because the HTML5 Sectioning Elements aren't mandatory to define an outline, to keep compatibility with the existing web dominated by HTML4, there is a way to define sections without them. This is called *implicit sectioning*.

The HTML Headings Elements (`<h1>` to `<h6>`) defines a new, implicit, section when they aren't the first heading of their parent, explicit, sections. The way this implicit section is positioned in the outline is defined by its relative rank with the previous heading in their parent section. If it is of a lower rank than the previous heading, it opens a implicit sub-section of the section. This code:

```
<section>
  <h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser known forest elephants.
    ...this section continues...
  <h3 class="implicit subsection">Habitat</h3>
  <p>Forest elephants do not live in trees but among them.
    ...this subsection continues...
</section>
```

leading to the following outline:

```
1. Forest elephants
   1.1 Habitat (implicitly defined by the h3 element)
```

If it is of the same rank as the previous heading, it closes the previous section (which may have been explicit!) and opens a new implicit one at the same level:

```
<section>
  <h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser known forest elephants.
    ...this section continues...
```

```
  <h1 class="implicit section">Mongolian gerbils</h1>
  <p>Mongolian gerbils are cute little mammals.
    ...this section continues...
</section>
```

leading to the following outline:

```
1. Forest elephants
2. Mongolian gerbils (implicitly defined by the h1 element, which closed the previous
```

If it is of a higher rank than the previous heading, it closes the previous section and opens a new implicit one at the higher level:

```
<body>
  <h1>Mammals</h1>
  <h2>Whales</h2>
  <p>In this section, we discuss the swimming whales.
    ...this section continues...
  <section>
    <h3>Forest elephants</h3>
    <p>In this section, we discuss the lesser known forest elephants.
      ...this section continues...
    <h3>Mongolian gerbils</h3>
      <p>Hordes of gerbils have spread their range far beyond Mongolia.
         ...this subsection continues...
    <h2>Reptiles</h2>
      <p>Reptiles are animals with cold blood.
          ...this subsection continues...
  </section>
</body>
```

leading to the following outline:

```
1. Mammals
   1.1 Whales (implicitly defined by the h2 element)
   1.2 Forest elephants (explicitly defined by the section element)
   1.3 Mongolian gerbils (implicitly defined by the h3 element, which closes the previ
   1.4 Reptiles (implicitly defined by the h2 element, which closes the previous sect
```

This is not the outline that one might expect by quickly glancing at the heading tags. To make your markup human-understandable, it is a good practice to use explicit tags for opening and closing sections, and to match the heading rank to the intended section nesting level. However, this is not required by the HTML5 specification. If you find that browsers are rendering your document outline in unexpected ways, check whether you have sections that are implicitly closed by heading elements.

An exception to the rule of thumb that heading rank should match the section nesting level is for sections that may be reused in multiple documents. For example, a section might be stored in a content-management system and assembled into documents at run time. In this case, a good practice is to start at `<h1>` for the top heading level of the reusable section. The nesting level of the reusable section will be determined by the section hierarchy of the document in which it appears. Explicit section tags are still helpful in this case.

Sectioning roots

A sectioning root is an HTML element that can have its own outline, but the sections and headings inside them do not contribute to the outline of their ancestor. Beside `<body>` which is the logical sectioning root of a document, these are often elements that introduce external content to the page: `<blockquote>`, `<details>`, `<fieldset>`, `<figure>` and `<td>`.

Example:

```
<section>
  <h1>Forest elephants</h1>
  <section>
    <h2>Introduction</h2>
    <p>In this section, we discuss the lesser known forest elephants</p>
  </section>
  <section>
    <h2>Habitat</h2>
    <p>Forest elephants do not live in trees but among them. Let's
       look what scientists are saying in "<cite>The Forest Elephant in Borneo</cite>'
    <blockquote>
       <h1>Borneo</h1>
       <p>The forest element lives in Borneo...</p>
    </blockquote>
  </section>
</section>
```

This example results in the following outline:

```
1. Forest elephants
   1.1 Introduction
   1.2 Habitat
```

This outline doesn't contain the internal outline of the `<blockquote>` element, which, being an external citation, is a sectioning root and isolates its internal outline.

Sections outside the outline

HTML5 introduces four new elements that allow defining sections that don't belong to the main outline of a web document:

1. The HTML Aside Section Element (`<aside>`) defines a section that, though related to the main element, doesn't belong to the main flow, like an explanation box or an advertisement. It has its own outline, but doesn't belong to the main one.
2. The HTML Navigational Section Element (`<nav>`) defines a section that contains navigation links. There can be several of them in a document, for example, one with page internal links, like a table of content, and another one with site navigational links. These links are not part of the main flow and outline and can be typically initially not rendered by screen reader and similar assistive technology.
3. The HTML Header Section Element (`<header>`) defines a page header, typically containing the logo and name of the site and possibly a horizontal menu. Despite its name, it is not necessarily positioned at the beginning of the page.
4. The HTML Footer Section Element (`<footer>`) defines a page footer, typically containing the copyright and legal noticed and sometimes some links. Despite its name, it is not necessarily positioned at the bottom of the page.

Addresses and publication time in sectioning elements

The author of a document often wants to publish some contact information, such the author's name and address. HTML4 allowed this via the `<address>` element, which has been extended in HTML5.

A document can be made of different sections from different authors. A section from another author than the one of the main page is defined using the `<article>` element. Consequently, the `<address>` element is now linked to its nearest `<body>` or `<article>` ancestor.

Similarly, the new HTML5 `<time>` element, with its `pubdate` boolean attribute set, represents the publication date associated to the whole document, respectively to the article, related to its nearest `<body>` or `<article>` ancestor.

Using HTML5 Elements in Non-HTML5 Browsers

Sections and headings elements should work in most non-HTML5 browsers. Though unsupported, they don't need a special DOM interface and they only need a specific CSS styling as unknown elements are styled as `display:inline` by default:

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

Of course the web developer can style them differently, but keep in mind that in a non-HTML5 browser, the default styling is different from what is expected for such elements. Also note that the `<time>` element has not been included, because the default styling for it in a non-HTML5 browser is the same as the one in an HTML5-compatible one.

This method has its limitation though, as some browsers do not allow styling of unsupported elements. That is the case of the Internet Explorer (version 8 and earlier), which need a specific script to allow this:

```
<!--[if lt IE 9]>
  <script>
    document.createElement("header" );
    document.createElement("footer" );
    document.createElement("section");
    document.createElement("aside"  );
    document.createElement("nav"    );
    document.createElement("article");
    document.createElement("hgroup" );
    document.createElement("time"   );
  </script>
<![endif]-->
```

This script means that, in the case of Internet Explorer (8 and earlier), scripting should be enabled in order to display HTML5 sectioning and headings elements properly. If not, they won't be displayed, which may be problematic as these elements are likely defining the structure of the whole page. That's why an explicit `<noscript>` element should be added for this case:

```
<noscript>
   <strong>Warning !</strong>
   Because your browser does not support HTML5, some elements are simulated using JSc
   Unfortunately your browser has disabled scripting. Please enable it in order to di
</noscript>
```

This leads to the following code to allow the support of the HTML5 sections and headings elements in non-HTML5 browsers, even for Internet Explorer (8 and older), with a proper fallback for the case where this latter browser is configured not to use scripting:

```
<!--[if lt IE 9]>
  <script>
    document.createElement("header" );
    document.createElement("footer" );
    document.createElement("section");
    document.createElement("aside"  );
    document.createElement("nav"    );
    document.createElement("article");
    document.createElement("hgroup" );
    document.createElement("time"   );
  </script>
  <noscript>
     <strong>Warning !</strong>
     Because your browser does not support HTML5, some elements are simulated using JS
     Unfortunately your browser has disabled scripting. Please enable it in order to d
  </noscript>
<![endif]-->
```

Conclusion

The new sections and headings elements introduced in HTML5 bring the ability to describe the structure and the outline of a web document in a standard way. They bring a big advantage for people having HTML5 browsers and needing the structure to help them understand the page, for instance people needing the help of some assistive technology. These new semantic elements are simple to use and, with very few burdens, can be made to work also in non-HTML5 browsers. Therefore they should be used without restrictions.

HTML5 Documentation

HTML	Audio/Video Canvas WebGL SVG MathML WebForms AppCache Microformats SemanticTags
JavaScript	Storage IndexedDB WebSockets WebWorkers Events Drag/Drop ProtocolHandler Geolocation Focus
CSS	NewSelectors Typography Visual Effects