Exercise 2b – HTML5 Markup

Objective

To investigate HTML5 structural tags and use them appropriately.

Overview

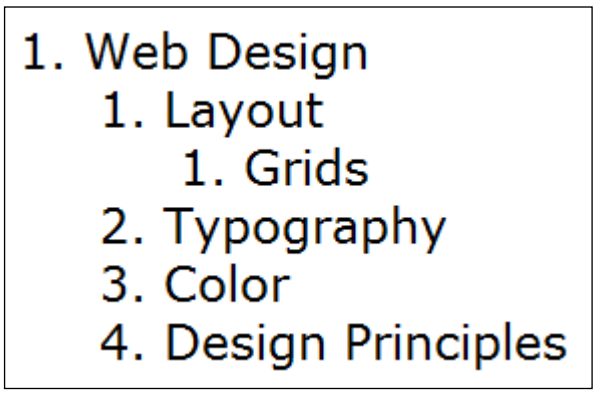
In this exercise, we will create a well-structured HTML5 document and learn to understand the HTML5 document outline. Understanding how information is structured is an incredibly important part of HTML5 knowledge as the informational hierarchy of the document carries an intent and structure that must make sense without the CSS. This exercise will take around **40 minutes**.

Part 1 – Examining XHTML implicit sectioning

1. From the starter folder for Exercise 2, open XHTMLHierarchy.html into any browser and take a look at the code. In this page as some appropriate use of the HTML. Note that we have one <h1> and a series of <h2> elements. The remaining markup belongs to one of these headings. We will start by investigating how this works.
2. Type the following URL into your browser:

```
http://gsnedders.html5.org/outliner/
```

3. Then copy and paste the source code into the HTML section of the page. And hit the 'Outline This!' button. You should then see a page outline like below:



4. Examine this in relation to the markup, you will notice the <h2> tags are all hierarchy children to the <h1> tag and that the <h3> Grids is a child of layout.
5. An outline created with heading content this way is said to consist of implicit, or implied, sections. Each heading creates its own implicit section, and any subsequent heading of a lower level starts another layer, or implicit sub-section, within it. An implicit section is ended by a heading of the same level or higher. In our example,

the “Layouts” section is ended by the beginning of the “Typography” section, and each section that contains details of an individual course is ended by the beginning of the next one.

6. Wrap the content in a tag and run it through an outliner, you will note it makes no difference to the outline. This is a very important HTML5 fact, the is a grouping element but is only a general divider with no semantic or structural/sectioning meaning. It is appropriate to use a to group content for stylistic purposes in HTML5, but we should aim to use new HTML5 semantic sectioning groups to apply more meaning to data.

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

Usage of for defining semantic sections, without defining specific values for the class attributes makes the automation of the outlining algorithm impossible ("Is that part of the outline of the page, defining a section or a subsection?" Or "is it only a presentational , only used for styling?").

In other terms, the HTML4 spec is very imprecise on what a section is and how its scope is defined. Automatic generation of outlines is important, especially for assistive technology that is likely to adapt the way they present information to the users according to the structure of the document.

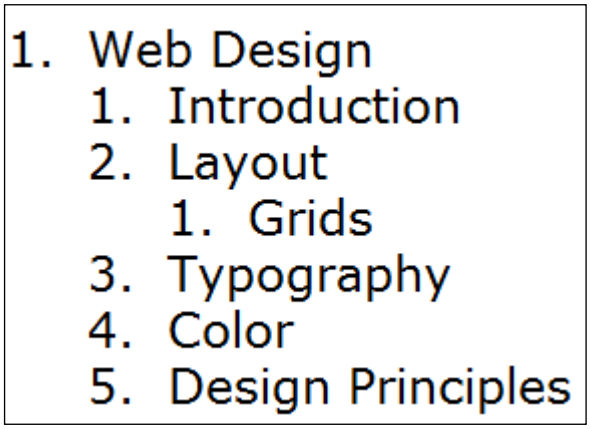
Part 2 – HTML5 Structural Element

In this section of the exercise, we will use the new structural elements and examine what change they make to the outlining algorithm. HTML5 removes the need for elements from the outlining algorithm by introducing a new element, <section>.

7. First of all, we will wrap the first <p> element (the one with the text some general info…) on the page in a <section> element. Save the page, view it in the browser. Nothing new or special happens. Copy the markup into the outliner and run it again. You will notice the text 1. Untitled Section. We have successfully sectioned the part of the document we want to work with, but the outliner is warning us it does not know what the section is about! For SEO, accessibility and better semantic identification of our data we want to stop any such problems, so we will put a <h2> with the text introduction as the first element in the section.
8. Run it through the outliner again and your outlook should look somewhat like this:

9. The outliner is now telling us that the <h1> is the first element in the document body. The <section> is no longer untitled, in fact, the algorithm has picked up that the <h2> is the identifier for the section. What we have learnt from this is that a section is a structural section element, where a is only a presentational element.

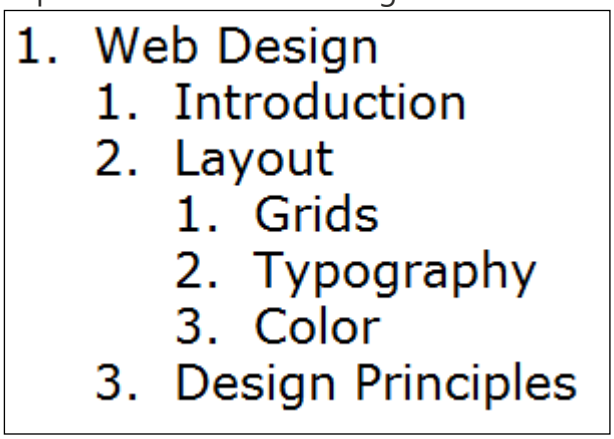
10. With that lesson learned, you need to adjust your markup with sections, until you get the following outline. As a hint, the <hn> elements form the structuring blocks.



11. The use of the <h2> is logical but would make no difference to the outline. If we used another element. Try changing the <h2> in the introduction element to a <h6> and check it in the outliner. No change. Remember, we are trying to build a logical document structure there not worrying about the appearance. Of course, if we check it in the browser it looks a little dumb, so switch it back to a <h2>.
12. We've also over used the <section> element, <sections> are meaningful linear sections of information. We want to rearrange the document to have 3 sections, Introduction, Layout and Design. Grids, Typography and Colour are to become <articles>.
13. To achieve this, you need to ensure there are only three sets of section elements, one around introduction. One from the start of layout until the end of the colour section. One surrounding the Design Principals area, but do ensure the last <p> element is outside the section.
14. We then need to surround the parts Grids, Typography and Colour elements in articles. The markup should look a little like this:

```
<div>
    <h1>
        Web Design</h1>
    <section>...</section>
    <section>
        <h2>
            Layout</h2>
        <p>
            Info about layouts</p>
        <article>...</article>
        <article>...</article>
        <article>...</article>
    </section>
    <section>...</section>
    <p>
        Where in the outline does this paragraph belong?</p>
</div>
```

15. Run the page through the outliner again and you should find the document looks as below. One of the marvellous things you will note is that even though the <h3> element would normally appear beneath the <h2> as a child, the other <h2> elements will not, by using <article> we have defined the semantic structure and importance of each heading.



16. 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.
17. With this in mind, I want to relate the <h1> and Introduction section into a single block. The <header> tag is perfect for this. Adjust your markup, so it looks like the following:

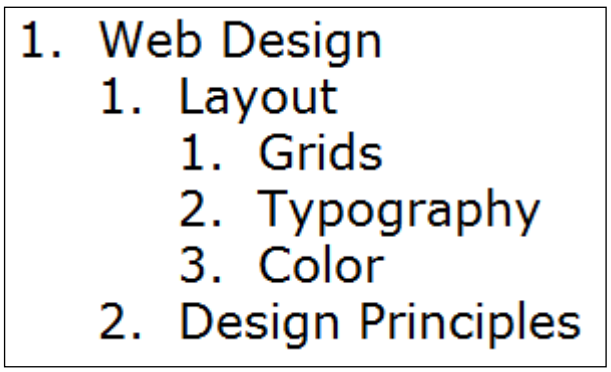
```
<header>
    <h1>
        Web Design</h1>
    <section>
        <h6>
            Introduction</h6>
        <p>
            Some general info about web design</p>
    </section>
</header>
```

18. Save the page and check it in the outliner, what has changed?

19. The outline did not change but the semantic value of the document has. The <header> element is the header of its nearest containing/sectioning element.
20. At this point, we can ask whether the <heading> really deserves a section element within. If the section was a separate unit of work, such as search functionality. If it is, and requires a heading, then it would remain as a section. As the document has developed, this doesn't quite seem right any more. We're going to alter the markup as shown below.

```
<header>
    <h1>Web Design</h1>
    <h2>Introduction to HTML5</h2>
    <div>
        <p>Some general info about web design</p>
    </div>
</header>
```

21. Save the page and run it through the outliner once again. Note that we have now turned that <h2> element into what is realistically a sub heading. However, the outliner is telling us the element is as important as the <h2> denoting a new section later in this document.
22. We want the <h2> where it is, but we want to take it out of the outline. Place the <h1> and <h2> in a <hgroup> element and run it through the outliner again.

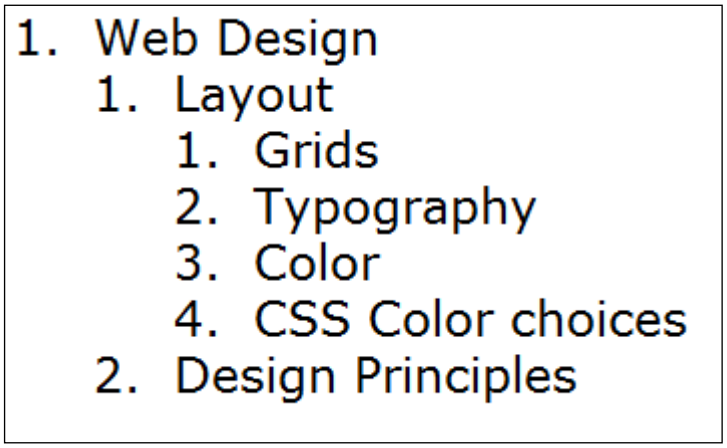


23. Note that the Introduction to HTML5 text has disappeared. By placing the headings in the <hgroup> tag, we have told the browser that the second header element is a semantic child of the preceding element it is not a sub branch in the document.

24. <aside> is one of the most interesting elements covering information that is tangential to the page i.e. the data expands or adds alternatives, but is not core to the understanding. So, let's add an example of this and see how this changes the outline. Add it after the colour article.

```
<aside>
    <h2>
        CSS Color choices</h2>
    <dl>
        <dt>RGB</dt>
        <dd>Using the rgb(r,g,b) css rule you can set color</dd>
        <dt>HEX</dt>
        <dd>The most common way of setting color</dd>
        <dt>HSL</dt>
        <dd>A new CSS3 color option</dd>
    </dl>
</aside>
```

25. If you review the outline you will note it is hierarchically equivalent to the <article> elements but semantically, we can note the different markup intent.



26. Time to turn our attentions to the bottom of the page and that spare <p> element and decide where it belongs. With the <h1> now in a <header> element, it seems to be an orphan element. If we open the page in Chrome and inspect the element, it tells us the page belongs to the outermost .

27. Although this is technically valid, it doesn't seem very useful as a general rule of thumb, if the is not the immediate parent of a semantic block or you are using it within a semantic block to apply styles to an element, it is probably in the wrong place.
28. What if we add the information into a <footer> element:

```
<footer>
    <p>Where in the outline does this paragraph belong?</p>
    <small>&copy; QA ltd</small>
</footer>
```

29. If you inspect the <p> element again, it will now be in a semantic container which is exactly what we want; plus if you run it through the outliner we have no Untitled Section either. The <footer> is always considered to adjoin to its relevant <header>.
30. We will finish off with the <nav> element, which we will place in just after the closing tag of the <header>.

```
<nav>
    <ul>
        <li><a href="#">Link 1</a></li>
        <li><a href="#">Link 2</a></li>
        <li><a href="#">Link 3</a></li>
    </ul>
</nav>
```

31. Save the page and check it in the outliner and you will note the return of the dreaded Untitled Section! Unlike the <footer>, we would normally place a heading here because the navigation is important enough to get a section of its own. This is a good rule of thumb, when the information is significant, it is a section and should be structured appropriately. We'll create a slightly more complex element

```
<nav>
    <h1>
        Site Navigation</h1>
    <section>
        <h2>
            My Links</h2>
        <ul>
            <li><a href="#">Link 1</a></li>
            <li><a href="#">Link 2</a></li>
            <li><a href="#">Link 3</a></li>
        </ul>
    </section>
             <section>
        <h2>
            Other Links</h2>
        <ul>
            <li><a href="#">Link 4</a></li>
            <li><a href="#">Link 5</a></li>
        </ul>
    </section>
</nav>
```

32. Save the page and test in the outliner it should look like the graphic below: