HTML 1: Overview

Chapter 2

Randy Connolly and Ricardo Hoar

Fundamentals of Web Development

© 2015 Pearson

http://www.funwebdev.com

Objectives

1. HTML Defined and its History
2. HTML Syntax
3. Semantic Markup
4. Structure of HTML
5. Quick Tour of HTML
6. HTML Semantic Elements

Section 1 of 6

HTML DEFINED + ITS HISTORY

Brief History of HTML

Did we mention that this will be brief?

- ARPANET of the late 1960s
- jump quickly to the first public specification of the HTML by Tim Berners-Lee in 1991
- HTML's codification by the World-Wide Web Consortium (better known as the **W3C**) in 1997.

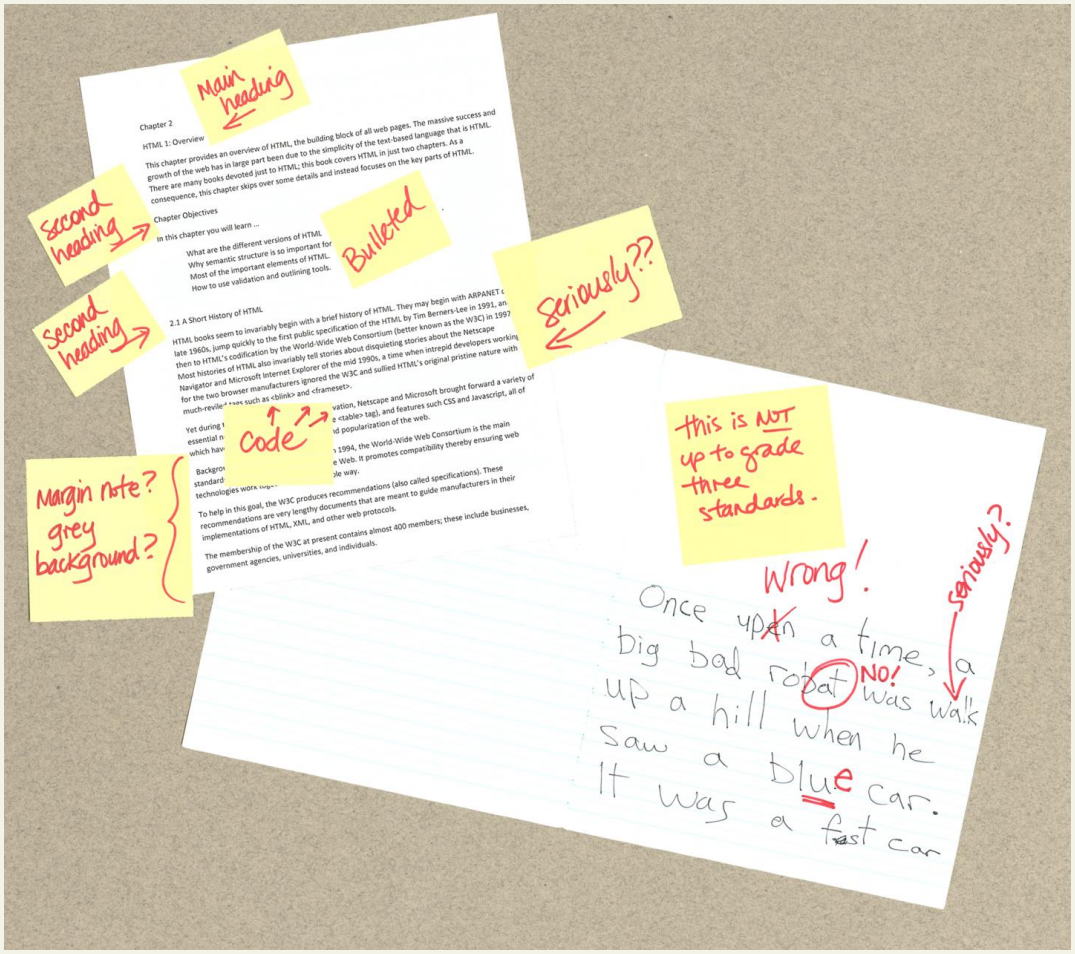
HTML Syntax

What is a markup language?

HTML is defined as a **markup language**.

- A markup language is simply a way of annotating a document in such a way to make the annotations distinct from the text being annotated.

Sample ad hoc markup



What is the W3C?

Standards

The W3C is the main standards organization for the World Wide Web.

To promotes compatibility the W3C produces **recommendations** (also called **specifications**).

In 1998, the W3C turned its attention to a new specification called XHTML 1.0, which was a version of HTML that used stricter XML (Extensible Markup Language) syntax rules.

XHTML

Partying like it's 1999

The goal of XHTML with its strict rules was to make page rendering more predictable by forcing web authors to create web pages without syntax errors.

XHTML

You too can be strict

The XML-based syntax rules for XHTML are pretty easy to follow.

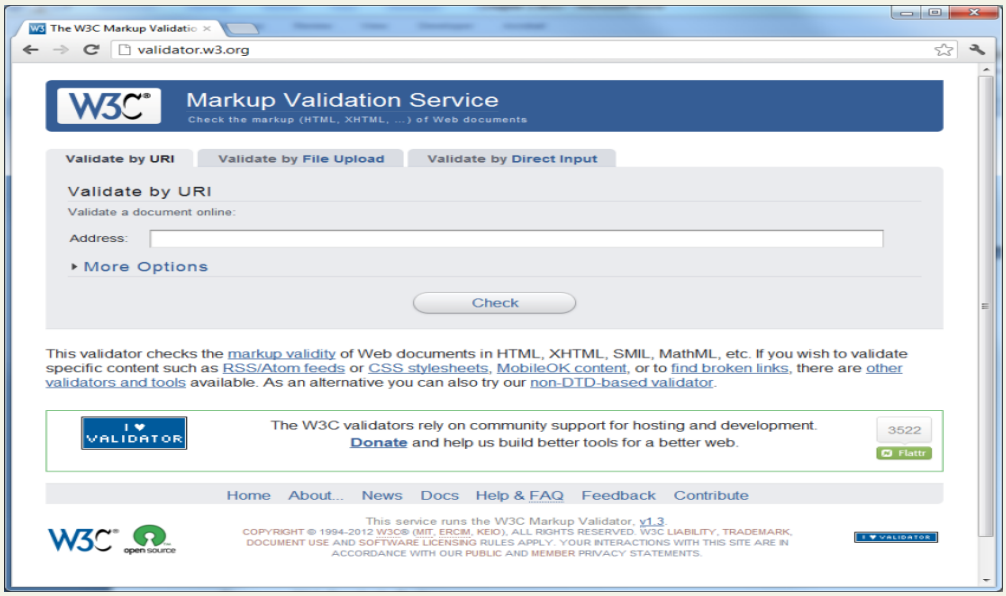
The main rules are:

- lowercase tag names,
- attributes always within quotes,
- and all elements must have a closing element (or be self-closing).

Validators

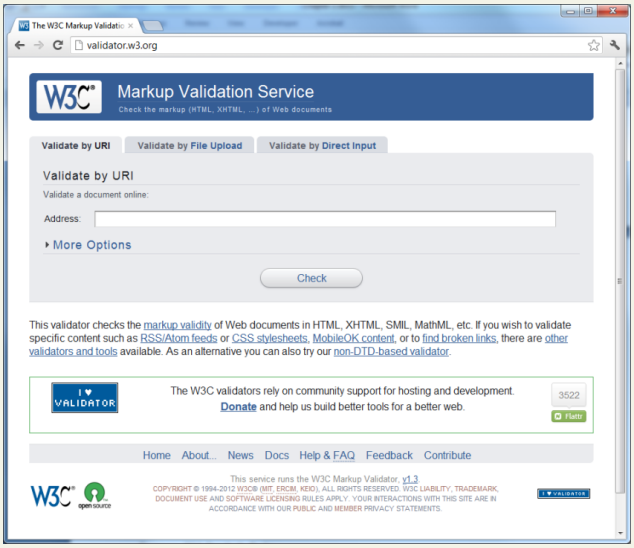
How to ensure your pages follow a standard

A key part of the standards movement in the web development community of the 2000s was the use of **HTML Validators** as a means of verifying that a web page's markup followed the rules for XHTML



How about an example

Only if you have an internet connection



Open a web browser to the W3C validator and find a few websites to test.

Type the URL into the bar, and you can check if the home page is valid against various standards (or auto-detect)

XHTML 2.0 and WHATWG

Where did it go?

In the mid 2000s, XHTML 2.0 proposed a revolutionary and substantial change to HTML.

- backwards compatibility with HTML and XHTML 1.0 was dropped.
- Browsers would become significantly less forgiving of invalid markup.

At around the same time, a group of developers at Opera and Mozilla formed the **WHATWG** (Web Hypertext Application Technology Working Group) group within the W3C.

This group was not convinced that the W3C's embrace of XML and its abandonment of backwards-compatibility was the best way forward for the web.

HTML5

The new hotness

By 2009, the W3C stopped work on XHTML 2.0 and instead adopted the work done by WHATWG and named it HTML5.

HTML5

Three main aims

There are three main aims to HTML5:

- Specify unambiguously how browsers should deal with invalid markup.
- Provide an open, non-proprietary programming framework (via Javascript) for creating rich web applications.
- Be backwards compatible with the existing web.

Section 2 of 6

HTML SYNTAX

Elements and Attributes

More syntax

HTML documents are composed of textual content and HTML elements.

An **HTML element** can contain text, other elements, or be empty. It is identified in the HTML document by tags.

HTML elements can also contain attributes. An **HTML attribute** is a name=value pair that provides more information about the HTML element.

In XHTML, attribute values had to be enclosed in quotes; in HTML5, the quotes are optional.

What HTML lets you do

- Insert images using the **<img>** tag
- Create links with the **<a>** tag
- Create lists with the **<ul>, ** and **<li>** tags
- Create headings with **<H1>, <H2>, …, <H6>**
- Define metatdata with **<meta>** tag
- And much more…

Elements and Attributes

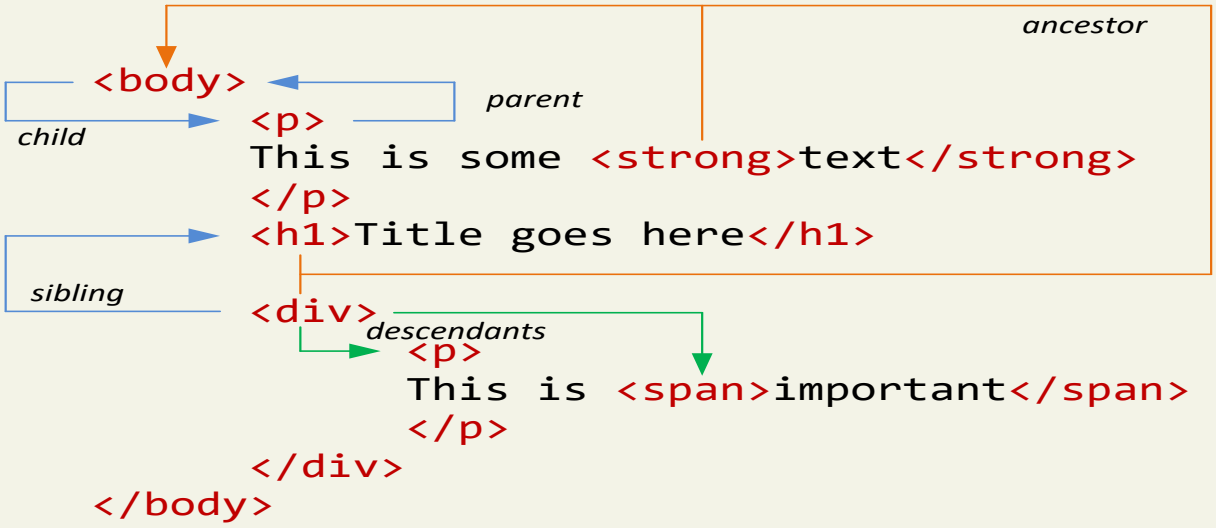
Nesting HTML elements

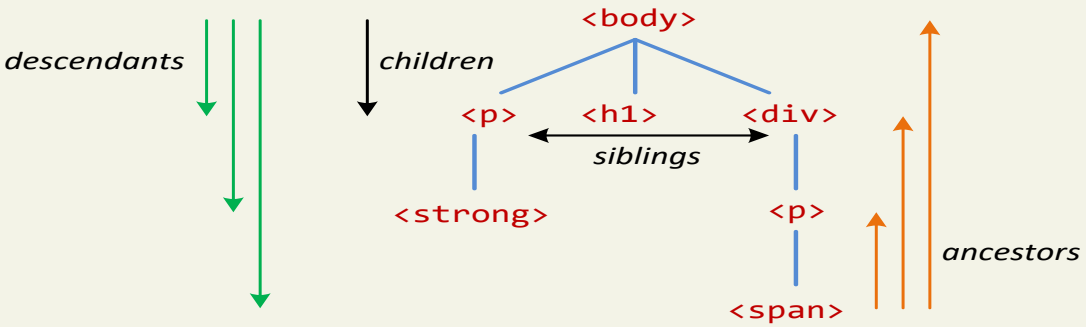
Often an HTML element will contain other HTML elements.

In such a case, the container element is said to be a parent of the contained, or child element.

Any elements contained within the child are said to be **descendents** of the parent element; likewise, any given child element, may have a variety of **ancestors**.

Hierarchy of elements



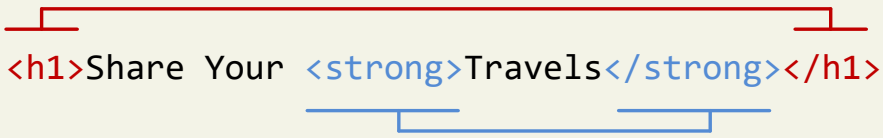


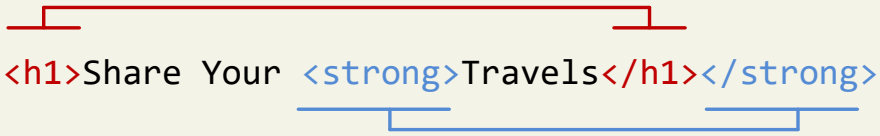
Nesting HTML elements

In order to properly construct a hierarchy of elements, your browser expects each HTML nested element to be properly nested.

That is, a child's ending tag must occur before its parent's ending tag.

Correct Nesting



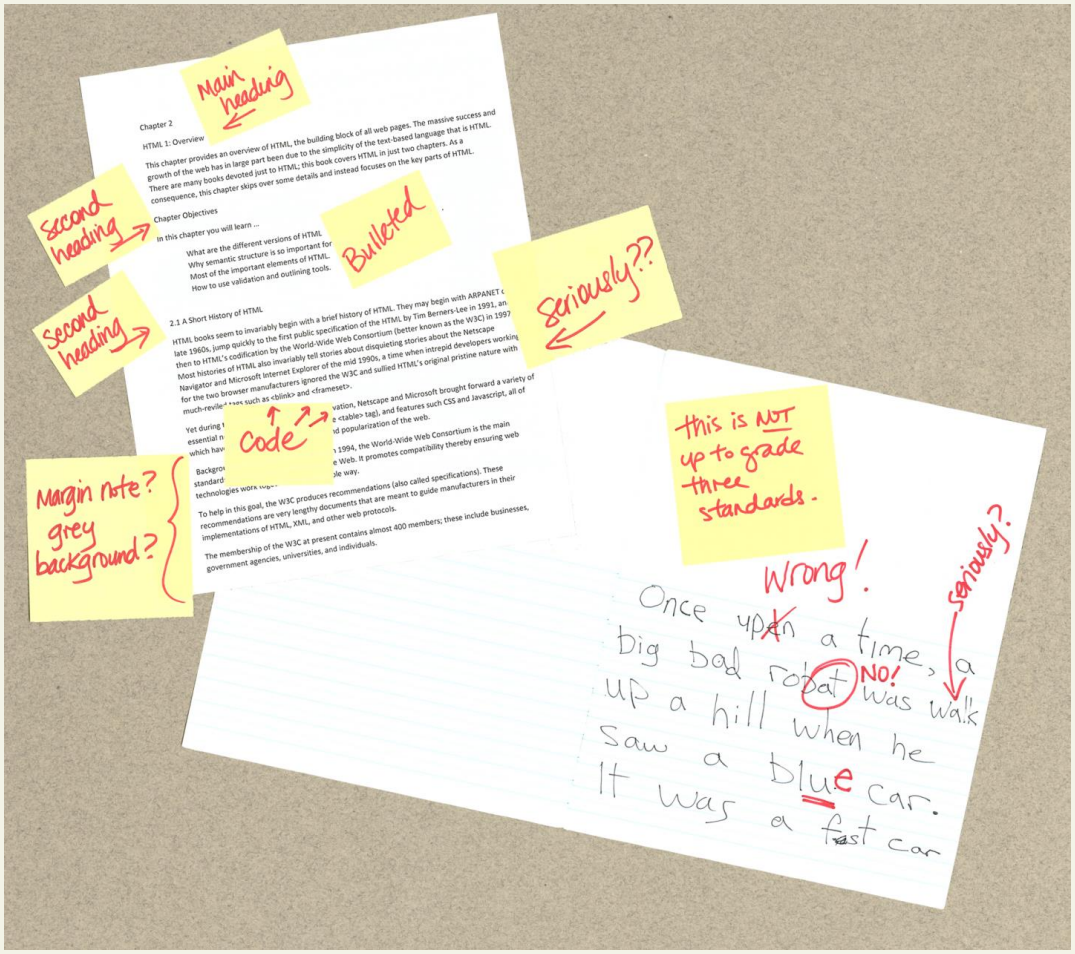


Incorrect Nesting

Section 3 of 6

SEMANTIC MARKUP

Sample ad hoc markup



Semantic Markup

What does it mean?

Over the past decade, a strong and broad consensus has grown around the belief that HTML documents should **only** focus on the structure of the document.

Information about how the content should look when it is displayed in the browser is best left to CSS (Cascading Style Sheets).

Semantic Markup

As a consequence, beginning HTML authors are often counseled to create **semantic HTML** documents.

That is, an HTML document should not describe how to visually present content, but only describe its content's structural semantics or meaning.

Semantic Markup

Its advantages

Eliminating presentation-oriented markup and writing semantic HTML markup has a variety of important advantages:

- **Maintainability**.
- **Faster**.
- **Accessibility**.
- **Search engine optimization**.

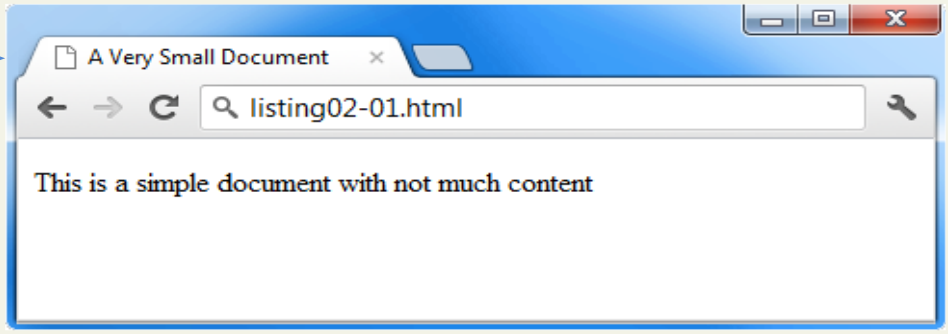
Section 4 of 6

STRUCTURE OF HTML

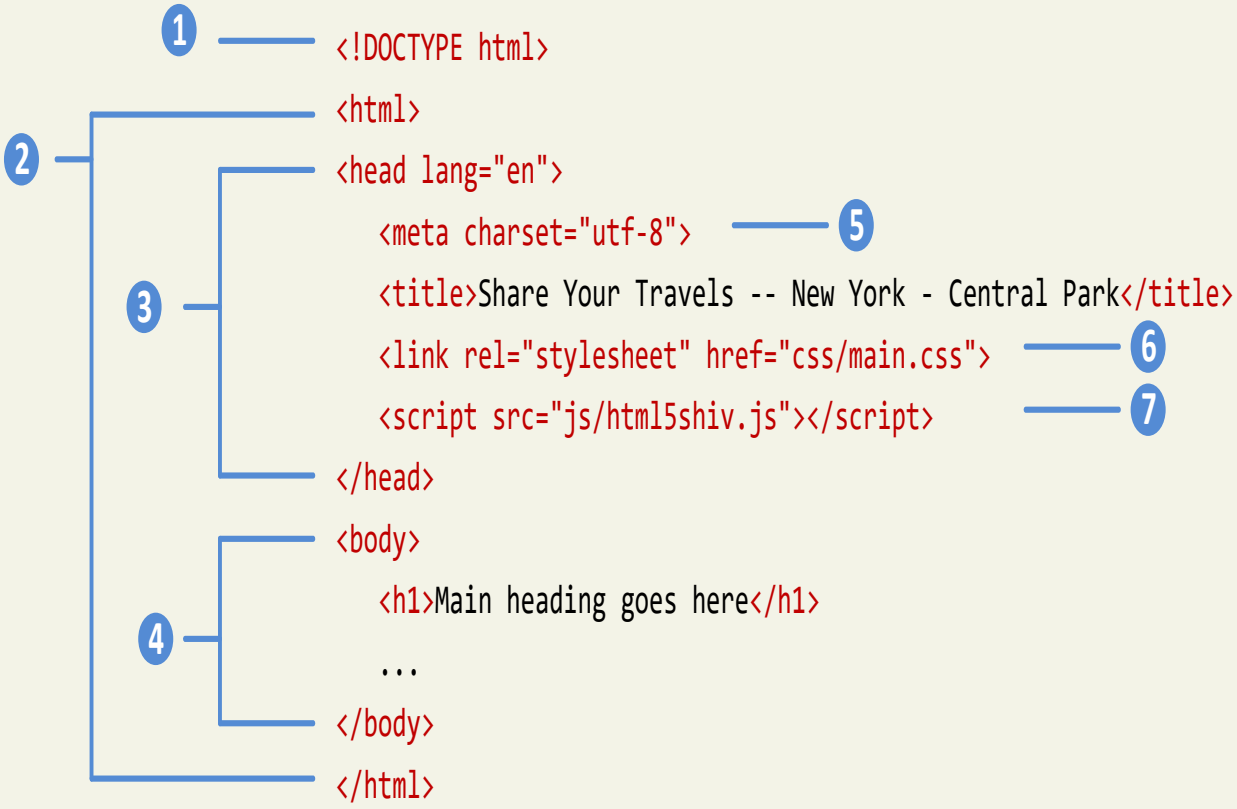
Simplest HTML document

1

```
<!DOCTYPE html>
<title>A Very Small Document</title>
<p>This is a simple document with not much content</p>
```



A more complete document

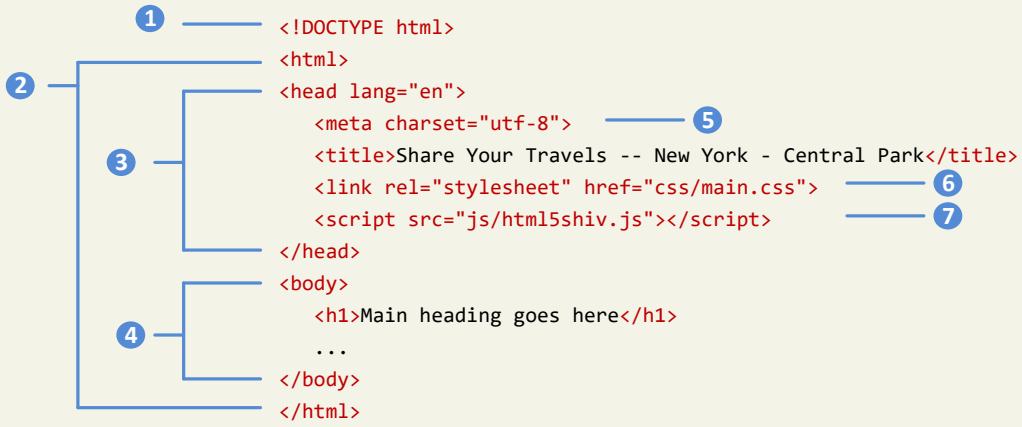


1 DOCTYPE

(short for **Document Type Definition**)

Tells the browser (or any other client software that is reading this HTML document) what type of document it is about to process.

Notice that it does not indicate what version of HTML is contained within the document: it only specifies that it contains HTML.



HTML, Head, and Body

HTML5 does not require the use of the <html>, <head>, and <body>.

However, in XHTML they were required, and most web authors continue to use them.

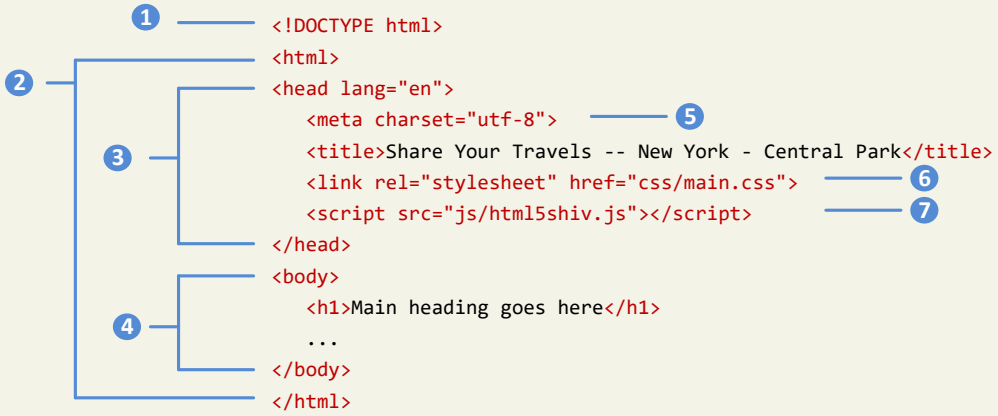
2. The <html> element is sometimes called the **root element** as it contains all the other HTML elements in the document.

Head and Body

HTML pages are divided into two sections: the **head** and the **body**, which correspond to the <head> and <body> elements.

3. The head contains descriptive elements *about* the document

4. The body contains content that will be displayed by the browser.

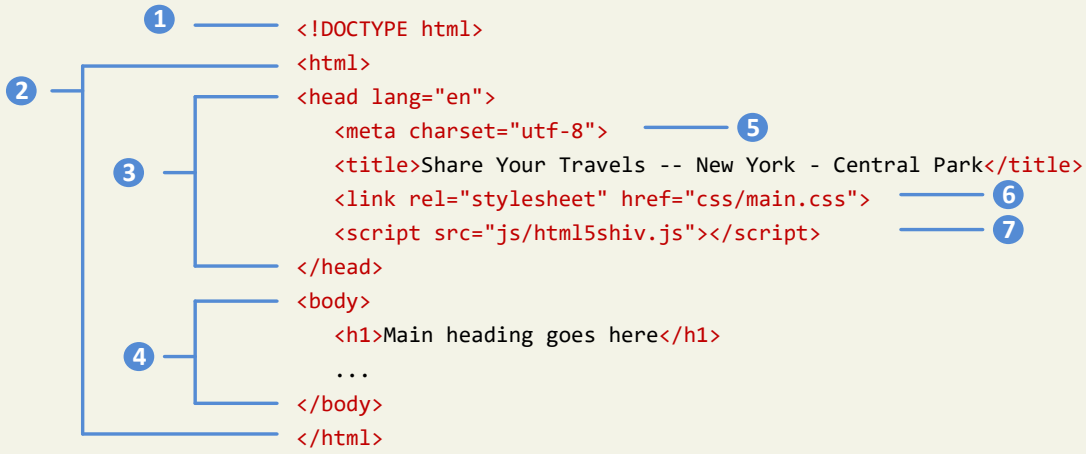


Inside the head

There are no brains

You will notice that the <head> element contains a variety of additional elements.

5. The first of these is the <meta> element. Our example declares that the character encoding for the document is UTF-8.



Inside the head

No brains but metas, styles and javascripts

6. Our example specifies an external CSS style sheet file that is used with this document.

7. It also references an external Javascript file.

Section 5 of 6

QUICK TOUR OF HTML

Why a quick tour?

HTML5 contains many structural and presentation elements – too many to completely cover in this presentation.

Rather than comprehensively cover all these elements, this presentation will provide a quick overview of the most common elements.

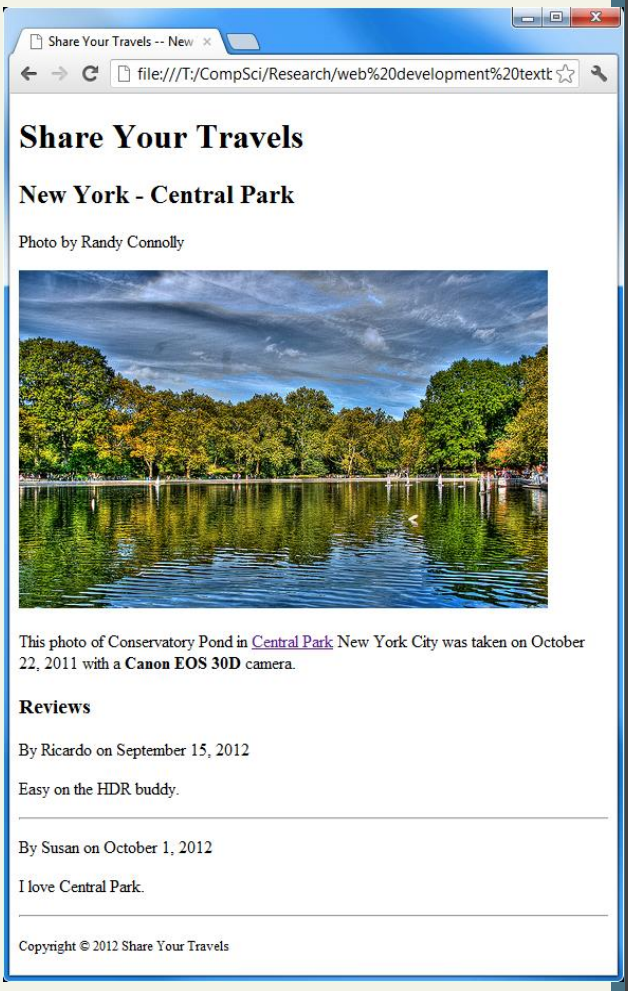
Sample Document

```
<body>
   <h1>Share Your Travels</h1>
   <h2>New York - Central Park</h2>
   <p>Photo by Randy Connolly</p>
   <p>This photo of Conservatory Pond in
      <a href="http://www.centralpark.com/">Central Park</a>
      New York City was taken on October 22, 2011 with a
      <strong>Canon EOS 30D</strong> camera.
   </p>
   <img src="images/central-park.jpg" alt="Central Park" />

   <h3>Reviews</h3>
   <div>
      <p>By Ricardo on <time>September 15, 2012</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>

   <div>
      <p>By Susan on <time>October 1, 2012</time></p>
      <p>I love Central Park.</p>
   </div>

   <p><small>Copyright &copy; 2012 Share Your Travels</small></p>
</body>
```

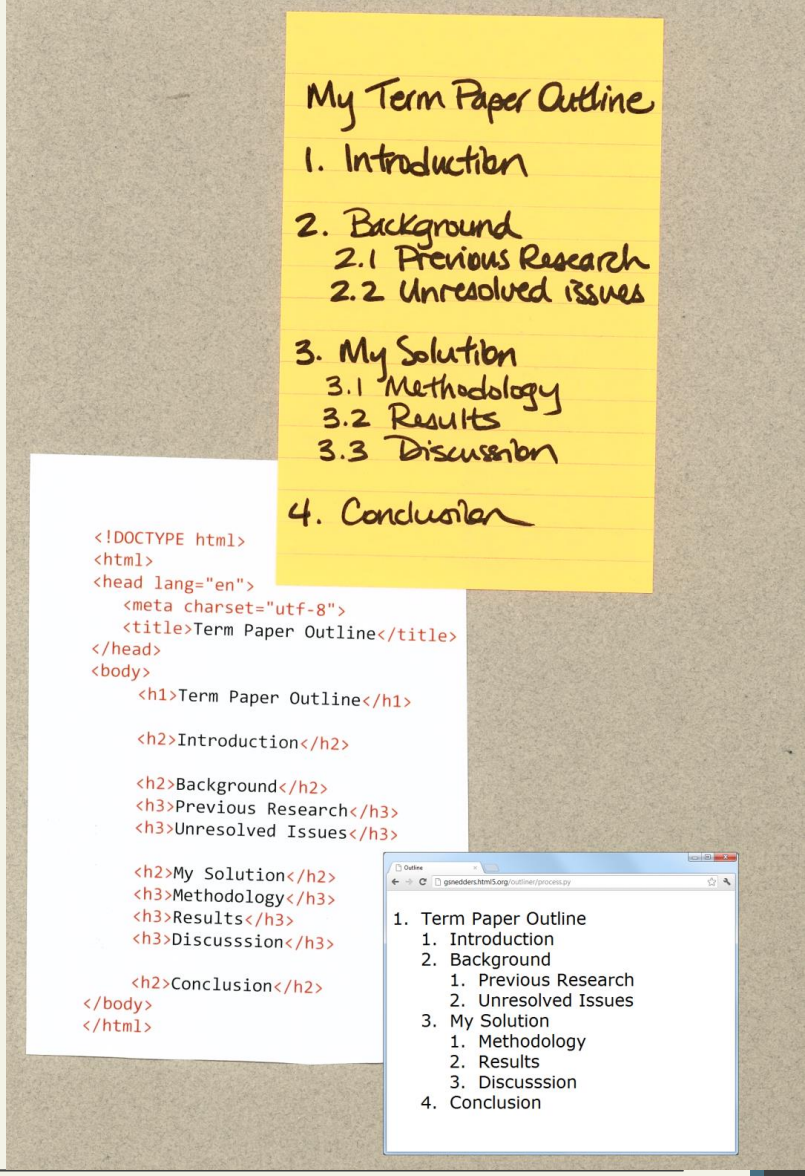


1 Headings

<h1>, <h2>, <h3>, etc

HTML provides six levels of heading (**h1, h2, h3**, …), with the higher heading number indicating a heading of less importance.

Headings are an essential way for document authors use to show their readers the structure of the document.



Headings

The browser has its own default styling for each heading level.

However, these are easily modified and customized via CSS.

2 Paragraphs

<p>

Paragraphs are the most basic unit of text in an HTML document.

Notice that the **<p>** tag is a container and can contain HTML and other **inline HTML elements**

inline HTML elements refers to HTML elements that do not cause a paragraph break but are part of the regular "flow" of the text.

6 Divisions

This ** ** tag is also a container element and is used to create a logical grouping of content

- The element has no intrinsic presentation.
- It is frequently used in contemporary CSS-based layouts to mark out sections.

3 Links

<a>

Links are created using the **<a>** element (the "a" stands for anchor).

A link has two main parts: the destination and the label.

```
<a href="http://www.centralpark.com">Central Park</a>
```

Destination

Label (text)

```
<a href="index.html"><img src="logo.gif" /></a>
```

Label (image)

Types of Links

You can use the anchor element to create a wide range of links:

- Links to external sites (or to individual resources such as images or movies on an external site).
- Links to other pages or resources within the current site.
- Links to other places within the current page.
- Links to particular locations on another page.
- Links that are instructions to the browser to start the user's email program.
- Links that are instructions to the browser to execute a Javascript function.

Different link destinations

Link to external site

```
<a href="http://www.centralpark.com">Central Park</a>
```

Link to resource on external site

```
<a href="http://www.centralpark.com/logo.gif">Central Park</a>
```

Link to another page on same site as this page

```
<a href="index.html">Home</a>
```

Link to another place on the same page

```
<a href="#top">Go to Top of Document</a>
```

Link to specific place on another page

```
<a href="productX.html#reviews">Reviews for product X</a>
```

Link to email

```
<a href="mailto://person@somewhere.com">Someone</a>
```

Link to javascript function

```
<a href="javascript://OpenAnnoyingPopup();">See This</a>
```

Link to telephone (automatically dials the number when user clicks on it using a smartphone browser)

```
<a href="tel:+18009220579">Call toll free (800) 922-0579</a>
```

URL Absolute Referencing

For external resources

When referencing a page or resource on an external site, a full **absolute reference** is required: that is,

- the protocol (typically, http://),
- the domain name,
- any paths, and then finally
- the file name of the desired resource.

URL Relative Referencing

An essential skill

We also need to be able to successfully reference files within our site.

This requires learning the syntax for so-called **relative referencing**.

When referencing a resource that is on the same server as your HTML document, then you can use briefer relative referencing. If the URL does not include the “http://” then the browser will request the current server for the file.

URL Relative Referencing

If all the resources for the site reside within the same **directory** (also referred to as a **folder**), then you can reference those other resources simply via their filename.

However, most real-world sites contain too many files to put them all within a single directory.

For these situations, a relative pathname is required along with the filename.

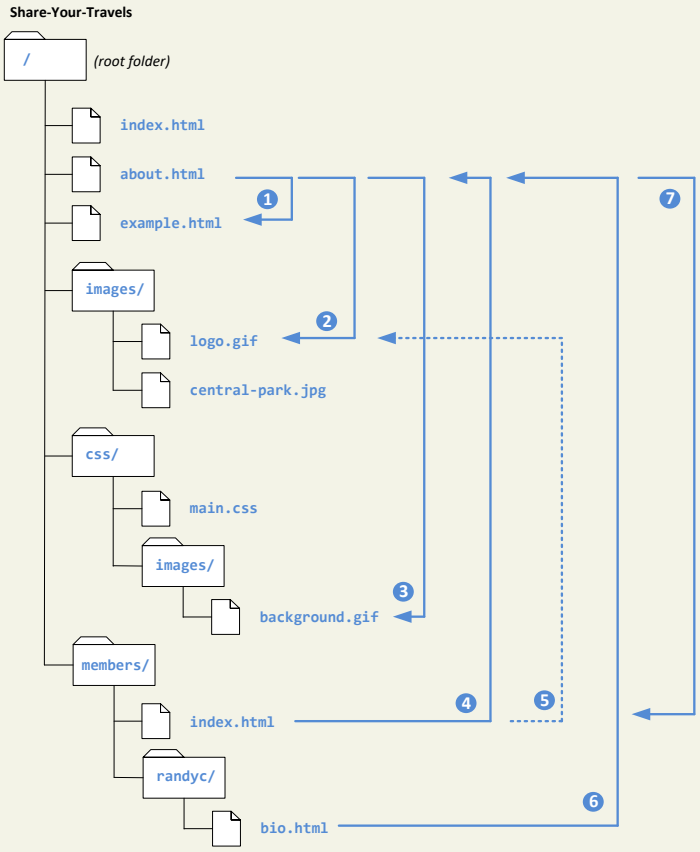
The **pathname** tells the browser where to locate the file on the server.

Pathnames

Pathnames on the web follow Unix conventions.

- Forward slashes ("/") are used to separate directory names from each other and from file names.
- Double-periods ("..") are used to reference a directory "above" the current one in the directory tree.

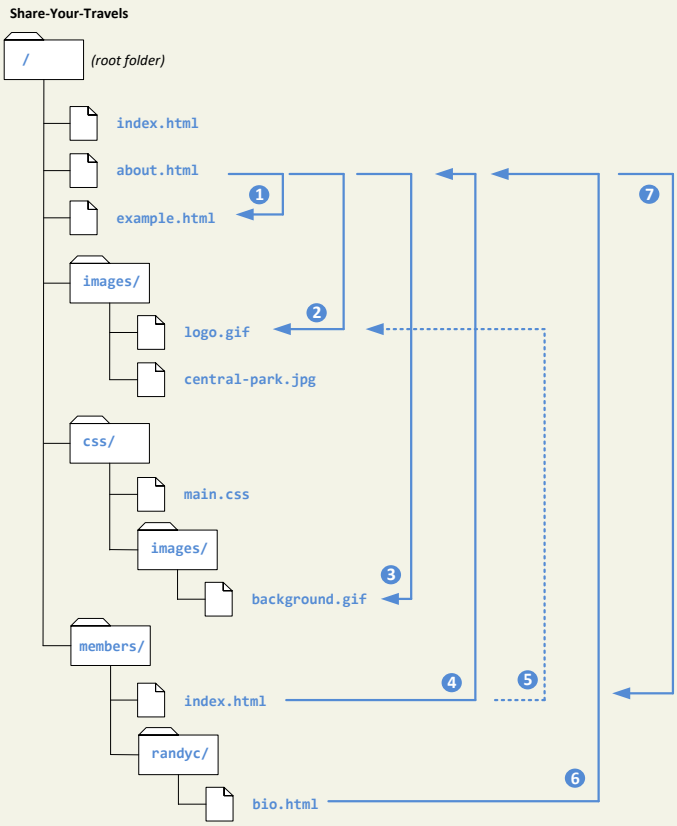
URL Relative Referencing



Relative Link Type	Example
1 Same Directory To link to a file within the same folder, simply use the file name.	To link to example.html from about.html (in Figure 2.18), use: `<a href="example.html">`
2 Child Directory To link to a file within a subdirectory, use the name of the subdirectory and a slash before the file name.	To link to logo.gif from about.html, use: `<a href="images/logo.gif">`
3 Grandchild/Descendant Directory To link to a file that is multiple subdirectories *below* the current one, construct the full path by including each subdirectory name (separated by slashes) before the file name.	To link to background.gif from about.html, use: `<a href="css/images/background.gif">`

4 Parent/Ancestor Directory
Use "../" to reference a folder *above* the current one. If trying to reference a file several levels above the current one, simply string together multiple "../".
	To link to about.html from index.html in members, use:
	`<a href="../about.html">`
	To link to about.html from bio.html, use:
	`<a href="../../about.html">`

URL Relative Referencing



5 Sibling Directory

Use "../"to move up to the appropriate level, and then use the same technique as for child or grandchild directories.

To link to logo.gif from index.html in members, use:

```
<a href="../images/about.html">
```

To link to background.gif from bio.html, use:

```
<a href="../../css/images/background.gif">
```

6 Root Reference

An alternative approach for ancestor and sibling references is to use the so-called **root reference** approach. In this approach, begin the reference with the root reference (the "/") and then use the same technique as for child or grandchild directories. **Note that these will only work on the server! That is, they will not work when you test it out on your local machine.**

To link to about.html from bio.html, use:

```
<a href="/about.html">
```

To link to background.gif from bio.html, use:

```
<a href="/images/background.gif">
```

7 Default Document

Web servers allow references to directory names without file names. In such a case, the web server will serve the default document, which is usually a file called index.html (apache) or default.html (IIS). **Again, this will only generally work on the web server.**

To link to index.html in members from about.html, use either:

```
<a href="members">
```

Or

```
<a href="/members">
```

Inline Text Elements

Do not disrupt the flow

Inline elements do not disrupt the flow of text (i.e., cause a line break).

HTML5 defines over 30 of these elements.

e.g., <a>, <br>, ,

Images

While the **<img>** tag is the oldest method for displaying an image, it is not the only way.

For purely decorative images, such as background gradients and patterns, logos, border art, and so on, it makes semantic sense to keep such images out of the markup and in CSS where they more rightly belong.

But when the images are content, such as in the images in a gallery or the image of a product in a product details page, then the <img> tag is the semantically appropriate approach.

Images

Specifies the URL of the image to display (note: uses standard relative referencing)

Text in title attribute will be displayed in a popup tool tip when user moves mouse over image.

```
<img src="images/central-park.jpg" alt="Central Park"  title="Central Park" width="80" height="40" />
```

Text in alt attribute provides a brief description of image's content for users who are unable to see it.

Specifies the width and height of image in pixels.

```
<!DOCTYPE html>
<html>
<body>

<h2>Image Size</h2>

<p>Here we specify the width and height of an image with the width and height
attributes:</p>

<img src="img_girl.jpg" alt="Girl in a jacket" width="500" height="600">

</body>
</html>
```

Image Size

Here we specify the width and height of an image with the width and height attributes:

Images

```
<!DOCTYPE html>
<html>
<body>

<p>If a browser cannot find the image, it will display the alternate text:</p>

<img src="wrongname.gif" alt="Flowers in Chania">

</body>
</html>
```

If a browser cannot find the image, it will display the alternate text:

Flowers in Chania

Lists

HTML provides three types of lists

Unordered lists. Collections of items in no particular order; these are by default rendered by the browser as a bulleted list.

Ordered lists. Collections of items that have a set order; these are by default rendered by the browser as a numbered list.

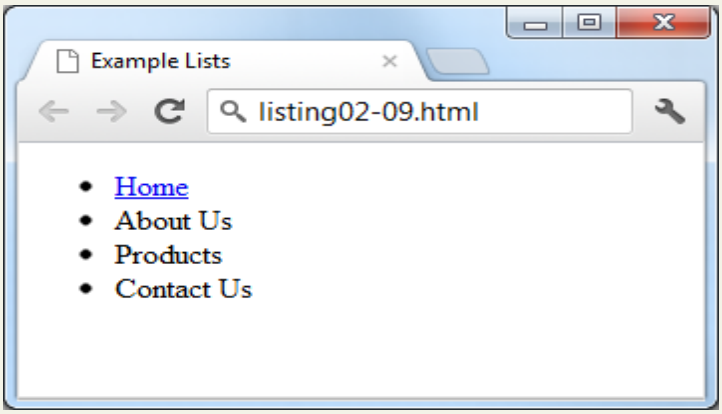
Definition lists. Collection of name and definition pairs. These tend to be used infrequently. Perhaps the most common example would be a FAQ list.

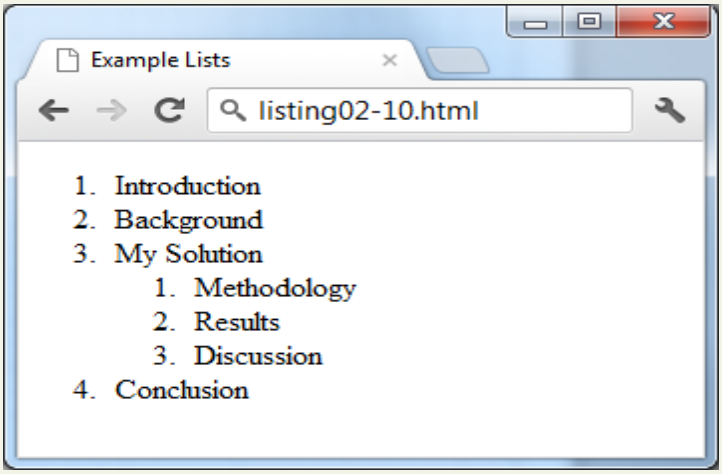
Lists

Notice that the list item element can contain other HTML elements

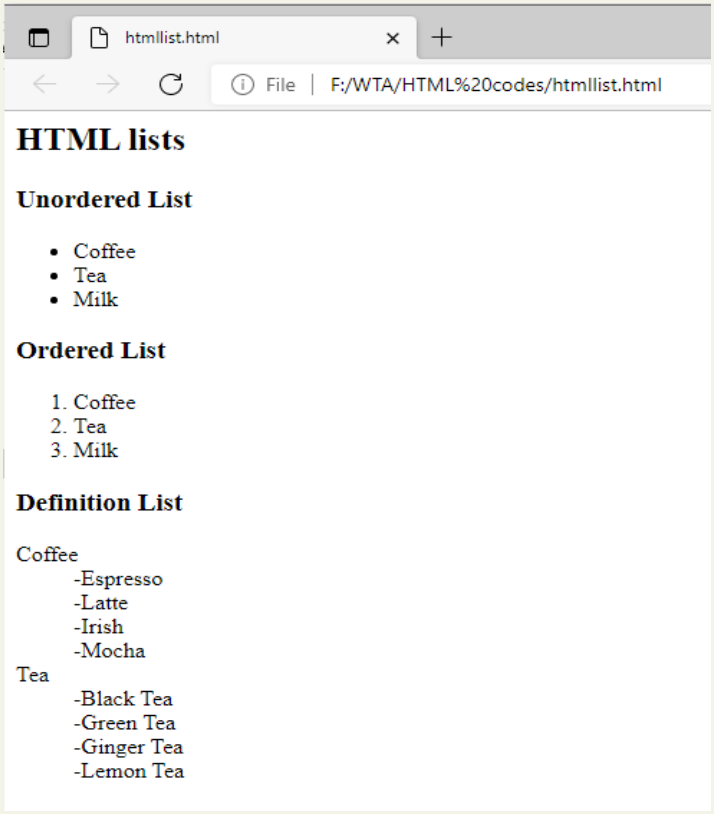
```
<ul>
   <li><a href="index.html">Home</a></li>
   <li>About Us</li>
   <li>Products</li>
   <li>Contact Us</li>
</ul>
```

```
<ol>
   <li>Introduction</li>
   <li>Background</li>
   <li>My Solution</li>
   <li>
      <ol>
         <li>Methodology</li>
         <li>Results</li>
         <li>Discussion</li>
      </ol>
   </li>
   <li>Conclusion</li>
</ol>
```





Lists




```
<html>
<body>
<h2>HTML lists</h2>
<h3> Unordered List </h2>
<ul>
   <li>Coffee</li>
   <li>Tea</li>
   <li>Milk</li>
</ul>
<h3> Ordered List </h2>
<ol>
   <li>Coffee</li>
   <li>Tea</li>
   <li>Milk</li>
</ol>
<h3> Definition List </h2>
<dl>
   <dt>Coffee</dt>
      <dd>-Espresso</dd>
      <dd>-Latte</dd>
      <dd>-Irish</dd>
      <dd>-Mocha</dd>
   <dt>Tea</dt>
      <dd>-Black Tea</dd>
   <dd>-Green Tea</dd>
      <dd>-Ginger Tea</dd>
      <dd>-Lemon Tea</dd>
</dl>
</body>
</html>
```

Character Entities

These are special characters for symbols for which there is either no way easy way to type in via a keyboard (such as the copyright symbol or accented characters) or which have a reserved meaning in HTML (for instance the “<” or “>” symbols).

They can be used in an HTML document by using the entity name or the entity number.

e.g., £ and ©

character entities

```
<!DOCTYPE html>
<html>
<body>

<h1>HTML Entity Example</h1>

<h2>The copyright sign: &copy;</h2>

</body>
</html>
```

HTML Entity Example

The copyright sign: ©

```
<!DOCTYPE html>
<html>
<body>

<h1>HTML Entity Example</h1>

<h2>The pound sign: &pound;</h2>

</body>
</html>
```

HTML Entity Example

The pound sign: £

character entities

¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
€	euro	&euro;	&#8364;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;

Section 6 of 6

HTML SEMANTIC ELEMENTS

HTML5 Semantic Elements

Why are they needed?

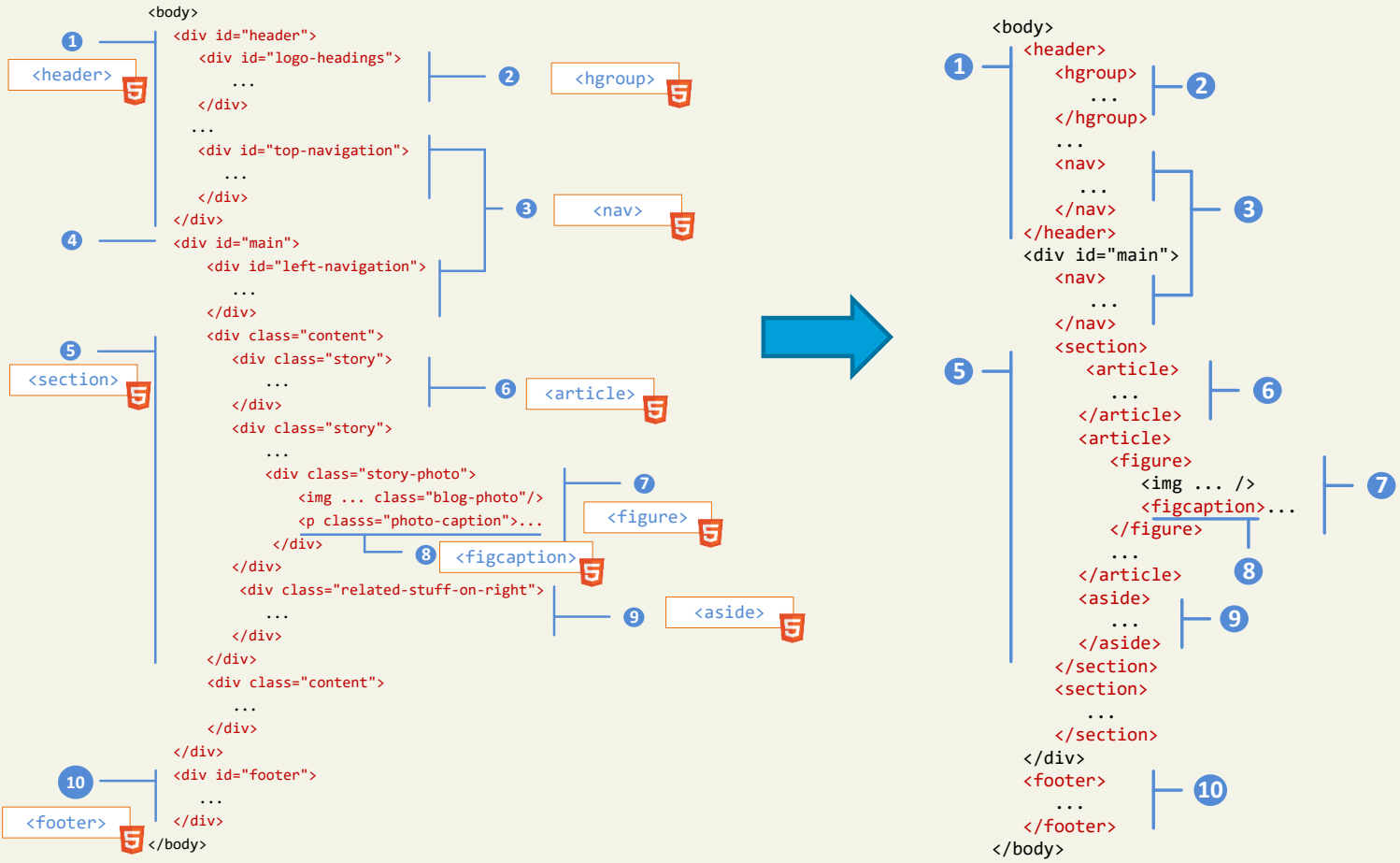
One substantial problem with modern, pre-HTML5 semantic markup:

> most complex web sites are absolutely packed solid with elements.

Unfortunately, all these elements can make the resulting markup confusing and hard to modify.

Developers typically try to bring some sense and order to the chaos by using id or class names that provide some clue as to their meaning.

XHTML versus HTML5



Header and Footer

<header> <footer>

Most web site pages have a recognizable header and footer section.

Typically the **header** contains

- the site logo
- title (and perhaps additional subtitles or taglines)
- horizontal navigation links, and
- perhaps one or two horizontal banners.

Header and Footer

<header> <footer>

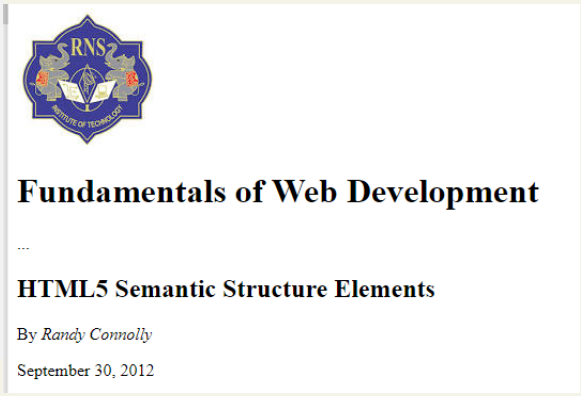
The typical footer contains less important material, such as

- smaller text versions of the navigation,
- copyright notices,
- information about the site's privacy policy, and
- perhaps twitter feeds or links to other social sites.

Header and Footer

Both the HTML5 <header> and <footer> element can be used not only for *page* headers and footers, they can also be used for header and footer elements within other HTML5 containers, such as <article> or <section>.

```
<header>
   <img src="logo.gif" alt="logo" />
   <h1>Fundamentals of Web Development</h1>
   ...
</header>
<article>
   <header>
       <h2>HTML5 Semantic Structure Elements </h2>
        <p>By <em>Randy Connolly</em></p>
        <p><time>September 30, 2012</time></p>
   </header>
   ...
</article>
```

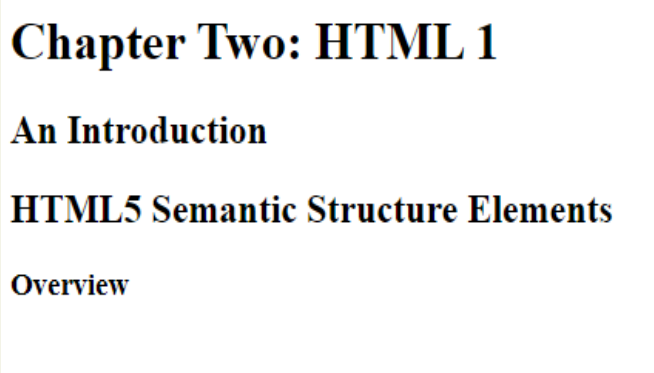


2 Heading Groups

<hgroup>

The <hgroup> element can be used to group related headings together within one container.

```
<header>
   <hgroup>
       <h1>Chapter Two: HTML 1</h1>
       <h2>An Introduction</h2>
   </hgroup>
</header>
<article>
   <hgroup>
      <h2>HTML5 Semantic Structure Elements </h2>
      <h3>Overview</h3>
   </hgroup>
</article>
```



3 Navigation

<nav>

The **<nav>** element represents a section of a page that contains links to other pages or to other parts within the same page.

Like the other new HTML5 semantic elements, the browser does not apply any special presentation to the <nav> element.

The <nav> element was intended to be used for major navigation blocks, presumably the global and secondary navigation systems.

Navigation

```
<!DOCTYPE html>
<html>
<body>

<h1>The nav element</h1>

<p>The nav element defines a set of navigation links:</p>

<nav>
<a href="/html/">HTML</a> |
<a href="/css/">CSS</a> |
<a href="/js/">JavaScript</a> |
<a href="/python/">Python</a>
</nav>

</body>
</html>
```

Articles and Sections

<article> <section>

The **<article>** element represents a **section of content** that forms an independent part of a document or site; for example, a magazine or newspaper article, or a blog entry.

The **<section>** element represents a **section of a document**, typically with a title or heading.

Articles

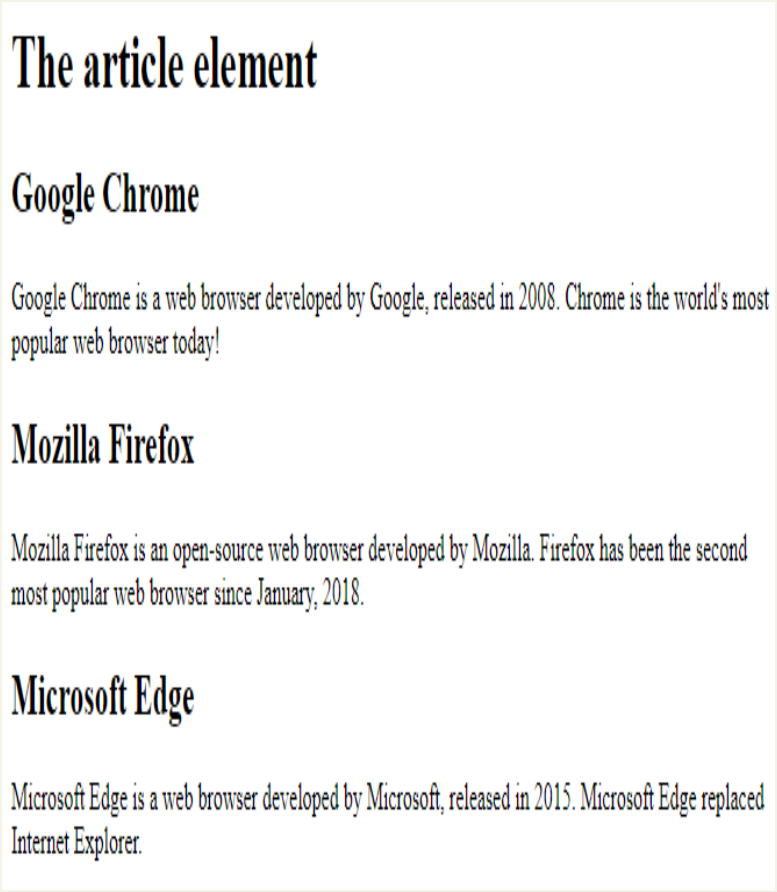
```
<!DOCTYPE html>
<html>
<body>

<h1>The article element</h1>

<article>
  <h2>Google Chrome</h2>
  <p>Google Chrome is a web browser developed by Google, released in 2008. Chrome is the world's most popular web browser today!</p>
</article>

<article>
  <h2>Mozilla Firefox</h2>
  <p>Mozilla Firefox is an open-source web browser developed by Mozilla. Firefox has been the second most popular web browser since January, 2018.
</p>
</article>

<article>
  <h2>Microsoft Edge</h2>
  <p>Microsoft Edge is a web browser developed by Microsoft, released in 2015. Microsoft Edge replaced Internet Explorer.</p>
</article>
```



Articles and Sections

According to the W3C, **<section>** is a much broader element, while the **<article>** element is to be used for blocks of content that could potentially be read or consumed independently of the other content on the page.

Figure and Figure Captions

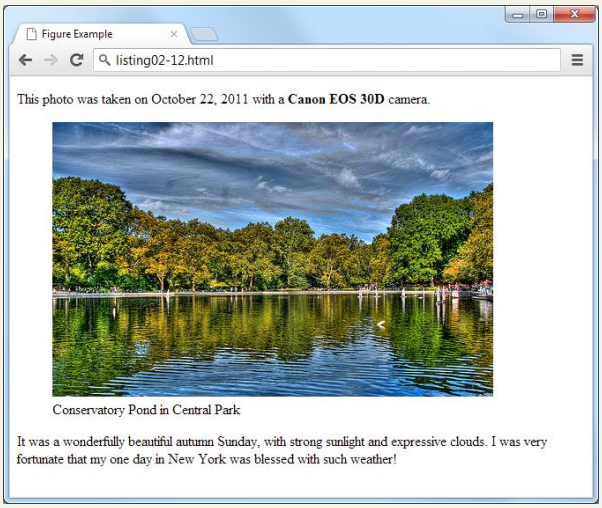
<figure> <figcaption>

The W3C Recommendation indicates that the <figure> element can be used not just for images but for any type of *essential* content that could be moved to a different location in the page or document and the rest of the document would still make sense.

Figure and Figure Captions

Figure could be moved to a different location in document …
But it has to exist in the document (i.e., the figure isn't optional)

```
<p>This photo was taken on October 22, 2011 with a Canon EOS 30D camera.</p>
<figure>
  <img src="images/central-park.jpg" alt="Central Park" /><br/>
  <figcaption>Conservatory Pond in Central Park</figcaption>
</figure>
<p>
It was a wonderfully beautiful autumn Sunday, with strong sunlight and
expressive clouds. I was very fortunate that my one day in New York was
blessed with such weather!
</p>
```



Aside

<aside>

The **<aside>** element is similar to the **<figure>** element in that it is used for marking up content that is separate from the main content on the page.

The **<aside>** element could thus be used for sidebars, pull quotes, groups of advertising images, or any other grouping of non-essential elements.

Aside

```
<!DOCTYPE html>
<html>
<body>

<h1>The aside element</h1>

<p>My family and I visited The Epcot center this summer. The weather was nice, and Epcot was amazing! I had a great summer together with my family!</p>

<aside>
  <h4>Epcot Center</h4>
  <p>Epcot is a theme park at Walt Disney World Resort featuring exciting attractions, international pavilions, award-winning fireworks and seasonal special events.</p>
</aside>

</body>
</html>
```

What You've Learned

1. HTML Defined and its History
2. HTML Syntax
3. Semantic Markup
4. Structure of HTML
5. Quick Tour of HTML
6. HTML Semantic Elements