Fundamentals of Web Development

Third Edition by Randy Connolly and Ricardo Hoar

Chapter 3

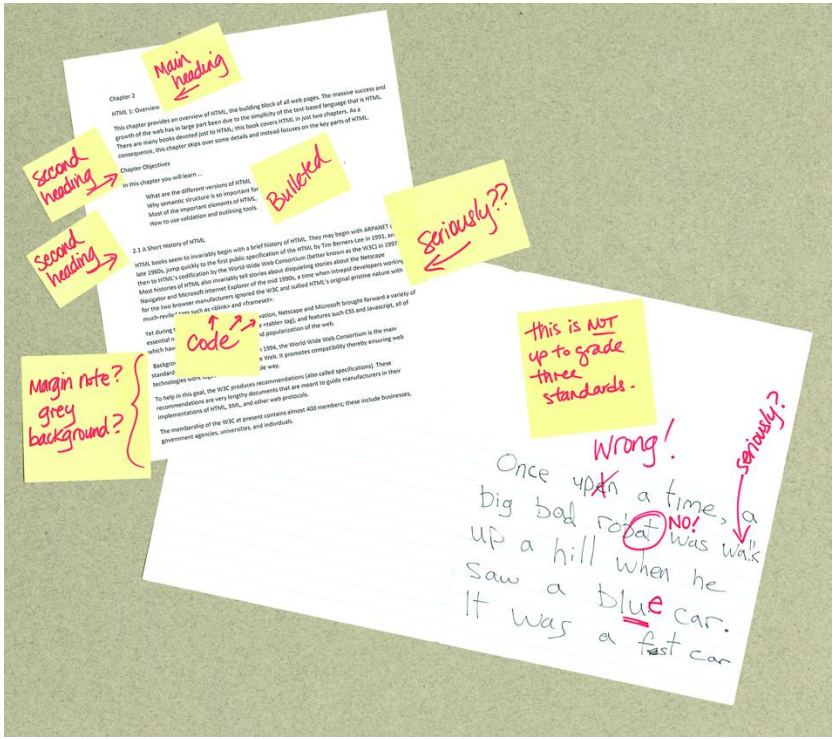
HTML 1: Introduction

In this chapter you will learn . . .

- A very brief history of HTML
- The syntax of HTML
- Why semantic structure is so important for HTML
- How HTML documents are structured
- A tour of the main elements in HTML
- The semantic structure elements in HTML5

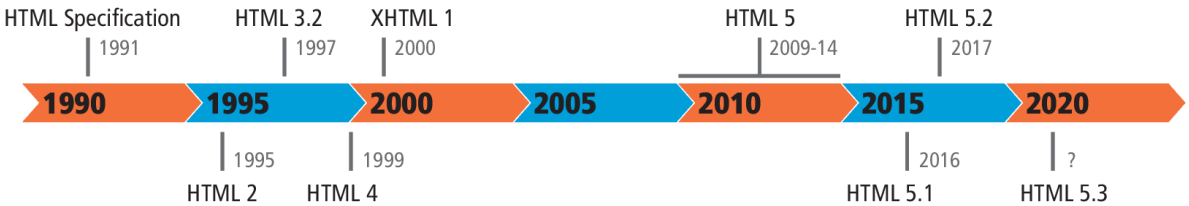
What Is HTML and Where Did It Come From?

- HTML is defined as a **markup language.** A markup language is simply a way of annotating a document in such a way as to make the annotations distinct from the text being annotated.
- You may very well have been the recipient of markup from caring parents or concerned teachers at various points in your past



HTML Markup Language

- At its simplest, **markup** is a way to indicate *information about the content* that is distinct from the content.
- This “information about content” in **HTML** is implemented via **tags** (or more formally, HTML elements, but more on that later).
- Since the initial HTML specification in 1991, HTML has gone through many interesting changes worth understanding in brief.



Early HTML

- Initial implementation of HTML and HTTP between 1990 and 1991 by Tim Berners-Lee and Robert Cailliau
- HTML's formal codification by the **World Wide Web Consortium** (better known as the **W3C**) between 1995 and 1997
- "browser wars" in the mid 1990s between Netscape Navigator and Microsoft Internet Explorer motivated many new tags and features such as CSS and JavaScript, but the development of new features happened quickly, and interoperability between browsers became a major issue
- In 1998 the W3C froze the HTML specification at version 4.01

XHTML

- In the late 1990s the W3C developed o a new specification called **XHTML 1.0**, which was a version of HTML that used stricter **XML**
- 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.
- To help web authors, two versions of XHTML were created:
  - XHTML 1.0 Strict and
  - XHTML 1.0 Transitional.
- Development on the XHTML 2.0 specification dragged on for many years

HTML5

- While the XHTML 2.0 specification was being developed, a small group of developers at Opera and Mozilla formed the **WHATWG** (Web Hypertext Application Technology Working Group).
- By 2009, the W3C stopped work on XHTML 2.0 and instead adopted the work done by WHATWG and named it **HTML5**.
- There are three main aims to HTML5:
  1. Specify unambiguously how browsers should deal with invalid markup.
  2. Provide an open, nonproprietary programming framework (via JavaScript) for creating rich web applications.
  3. Be backward compatible with the existing web.

HTML Syntax

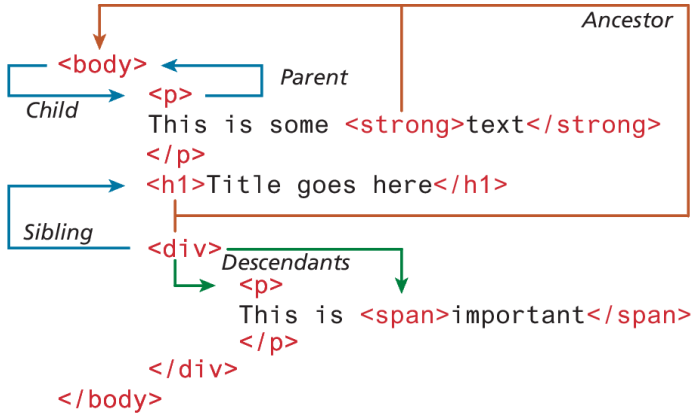
- **HTML documents** are composed of **content** and HTML elements.
- An **HTML element** is identified in the HTML document by tags.
  - A **tag** consists of the element name within angle brackets.
- The **element name** appears in both the **opening tag** and the **closing tag**.
- HTML elements can also contain attributes. An **HTML attribute** is a name=value pair that provides more information about the element

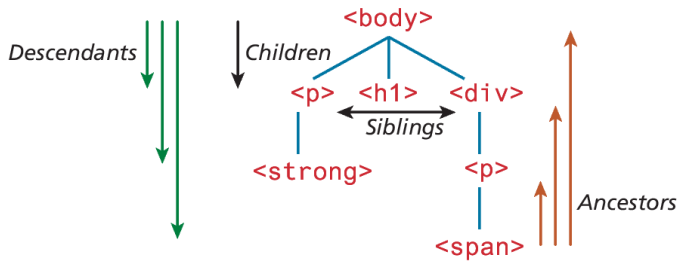
Empty Element

- An **empty element** does not contain any text content; instead, it is an instruction to the browser to do something.
- Perhaps the most common empty element is <img>, the image element.
- In XHTML, empty elements had to be terminated by a trailing slash (as shown in image). In HTML5, the trailing slash in empty elements is optional.

Nesting HTML Elements

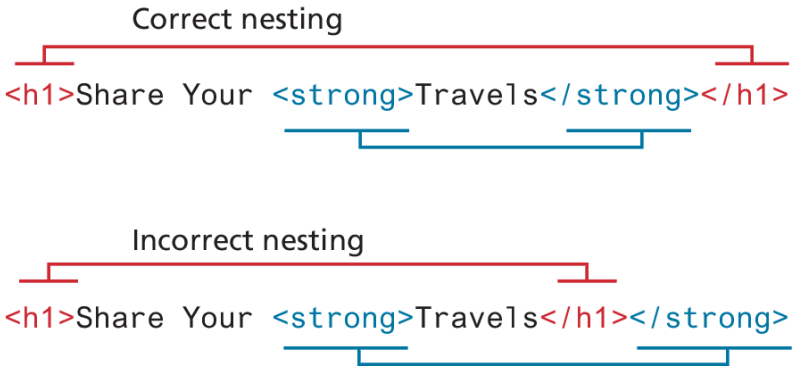
- Often an HTML element will contain other HTML elements. In such a case, the container element is said to be a **parent** of the contained, or **child**, element.
- Any elements contained within the child are said to be **descendants** of the parent element; likewise, any given child element may have a variety of **ancestors**.
- This concept is called the **Document Object Model** (DOM) formally, though for now we will only refer to its hierarchical aspects.





Correct Nesting

- In order to properly construct this hierarchy of elements, your browser expects each HTML nested element to be properly nested.
- A child's ending tag must occur before its parent's ending tag



Semantic Markup

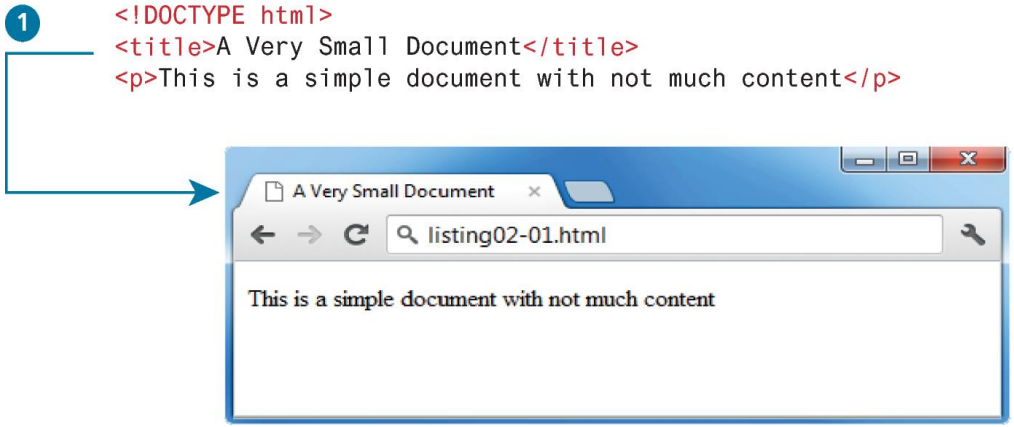
- HTML documents should **only** focus on the structure of the document
- Information about how the content should look is best left to CSS (Cascading Style Sheets), a topic introduced in the next chapter
- As a consequence, beginning HTML authors are often counseled to create **semantic HTML** documents.
- HTML document should not describe how to visually present content but only describe its content's structural semantics or meaning

Semantic Markup Advantages

- **Maintainability**. Semantic markup is easier to update and change than web pages that contain a great deal of presentation markup.
- **Performance**. Semantic web pages are typically quicker to author and faster to download.
- **Accessibility**. Not all web users are able to view the content on web pages. Users with sight disabilities experience the web using voice-reading software.
- **Search engine optimization**. For many site owners, the most important users of a website are the various search engine crawlers.

Structure of HTML Documents

- Figure 3.8 illustrates one of the simplest *valid* HTML5 documents you can create.



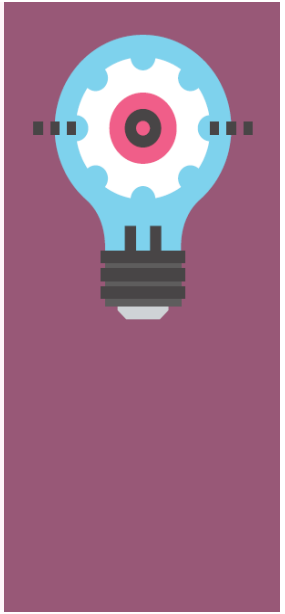
The title element

PROTIP

The **<title>** element plays an important role in search engine optimization (SEO), that is, improving a page's rank (its position in the results page after a search) .

While each search engine uses different algorithms for determining a page's rank, the **title** (and the major headings) provides a key role in determining what a given page is about.

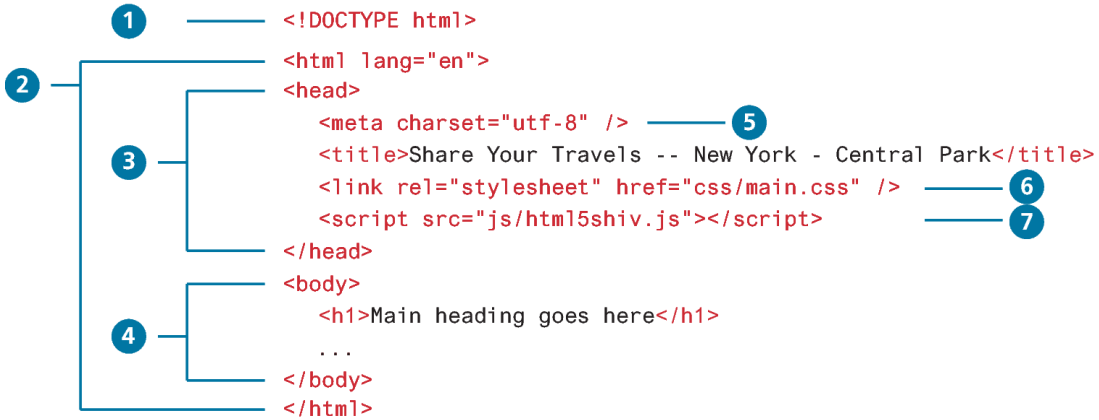
As a result, be sure that a page's title text briefly summarizes the document's content. As well, put the most important content first in the title. Most browsers limit the length of the title that is displayed in the tab or window title to about 60 characters. Chapter 18 goes into far greater detail on SEO.



Structure elements of an HTML5 document

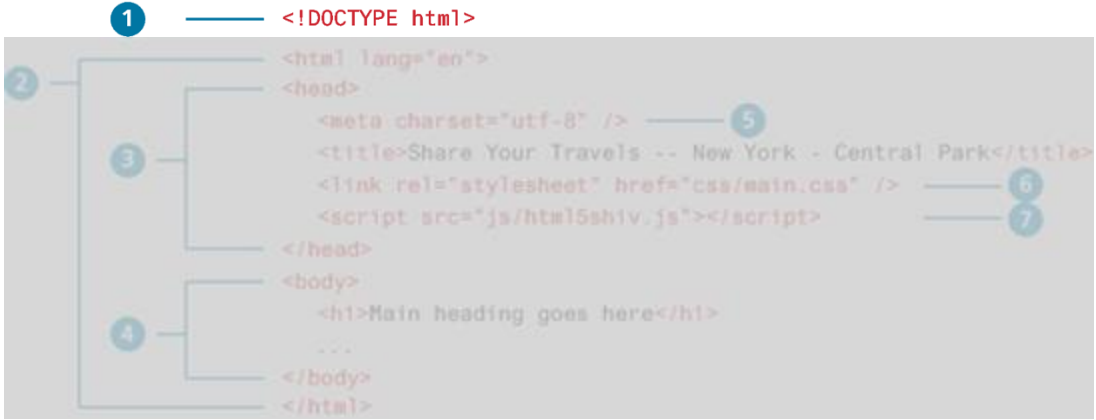
- Consider this more complete HTML5 document that includes structural elements as well as some other common HTML elements.

- Let's explore this page in detail



DOCTYPE

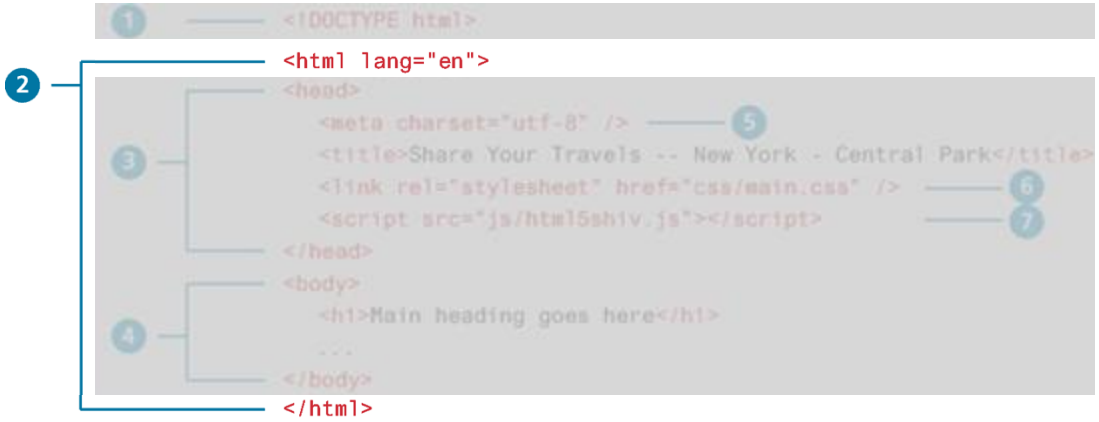
- The DOCTYPE declaration tells the browser what type of document it is about to process.
- It does not indicate what version of HTML is contained within the document



<Html> element

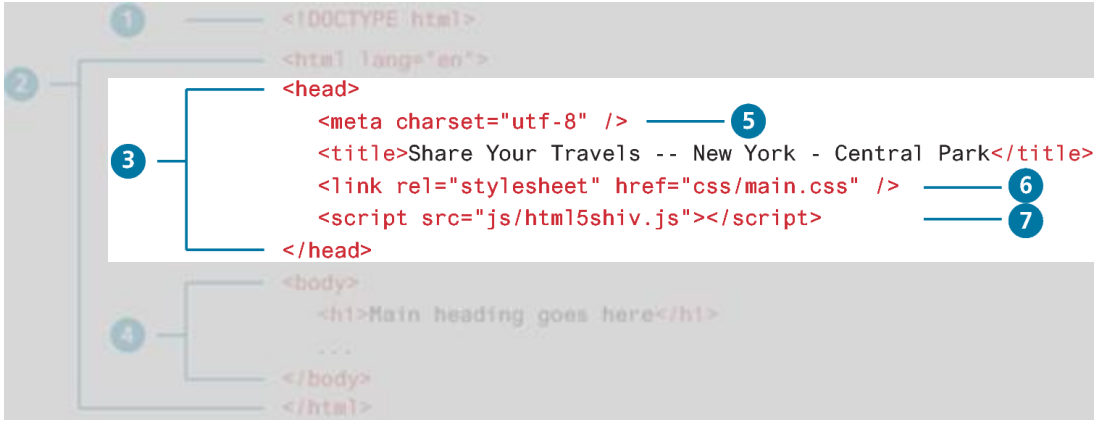
HTML5 does not require the use of the <html>, <head>, and <body> elements. However, in XHTML they were required, and most web authors continue to use them.

The <html> element is sometimes called the **root element** as it contains all the other HTML elements in the document. The optional **lang** attribute tells the browser the language that is being used.



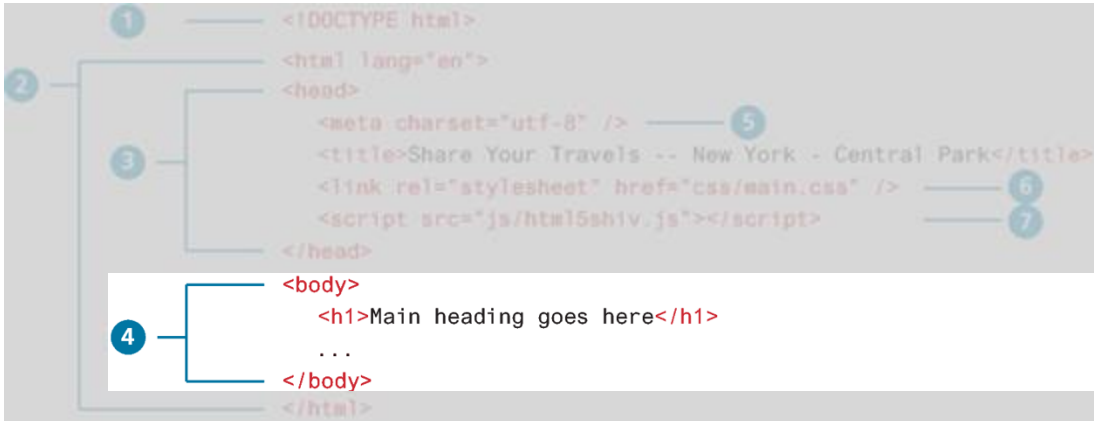
<Head> Element

The **head** contains descriptive elements *about* the document, such as its title, encoding, and any style sheets or JavaScript files it uses.



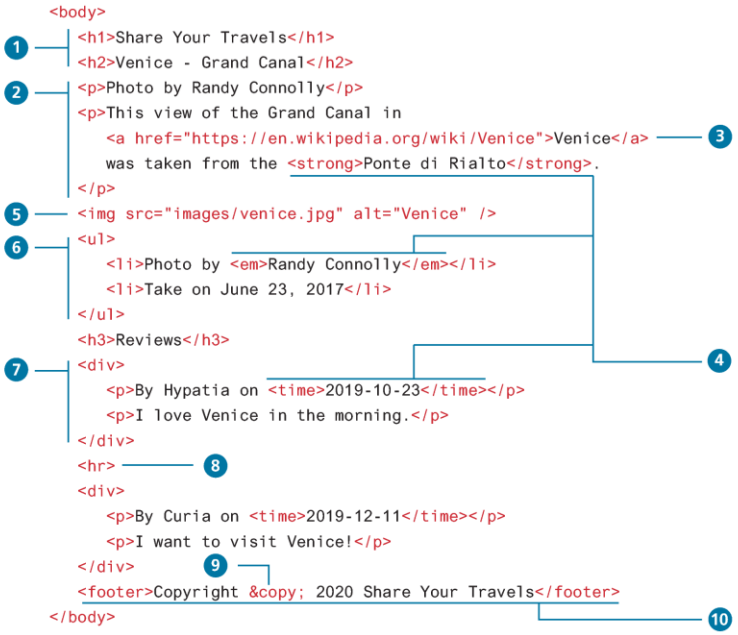
<Body> Element

The body contains content (both HTML elements and regular text) that will be displayed by the browser. The rest of this chapter and the next chapter will cover the HTML that will appear within the body.



Quick Tour of HTML Elements

1. **Headings**. Describes the main structure of document. There are six levels of headings.
2. **Paragraphs**. The basic unit of text in HTML. As block-level elements, browsers typically add newlines before and after the element.
3. **Link**. Hyperlinks are essential feature of all web pages and can reference another page or another location in same page.
4. **Inline Text Elements**. These do not change the flow of text and provide more information about text.
5. **Image**. Used to display an image by specifying a filename or URL.



Quick Tour of HTML Elements (cont)

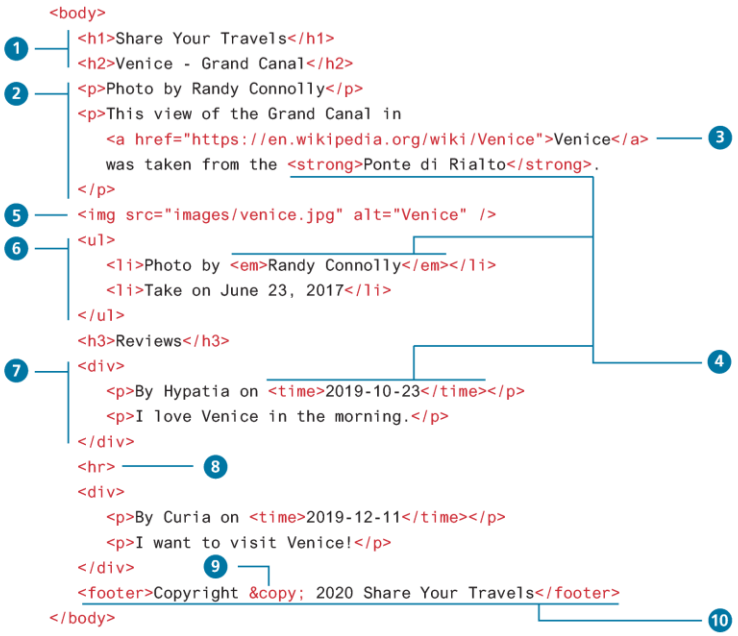
6. **Unordered List**. Used to display a bulleted list. Within a list is a collection of list item elements.

7. **Division**. Container for text or other HTML elements. Like paragraphs, they are also block-level elements.

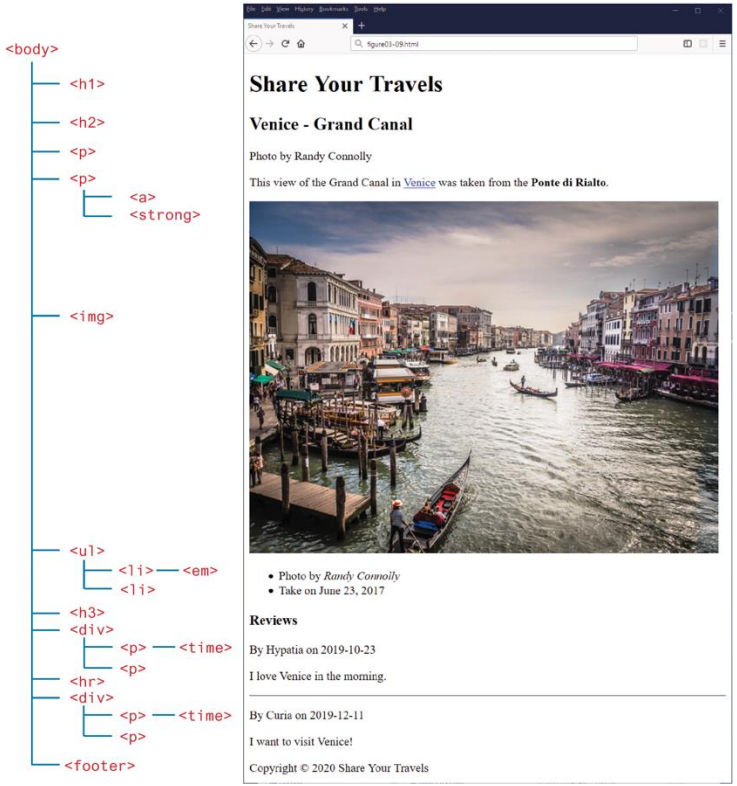
8. **Horizontal Rule**. Indicates a thematic break in the text. Usually displayed as a horizontal line.

9. **Character Entity**. The mechanism for including special symbols (such as ©) or characters that have a reserved meaning in HTML.

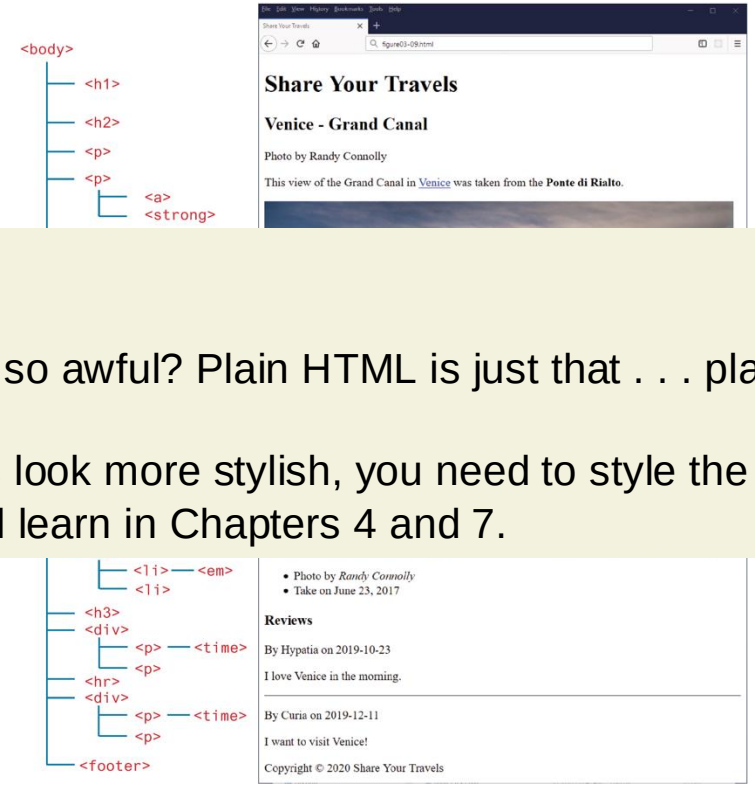
10. **Semantic Block Element**. Special containers in HTML5 for describing structural elements in a document.



In the browser



In the browser (note)



NOTE

Why does this look so awful? Plain HTML is just that . . . plain looking.

To make our pages look more stylish, you need to style the elements using CSS, which you will learn in Chapters 4 and 7.

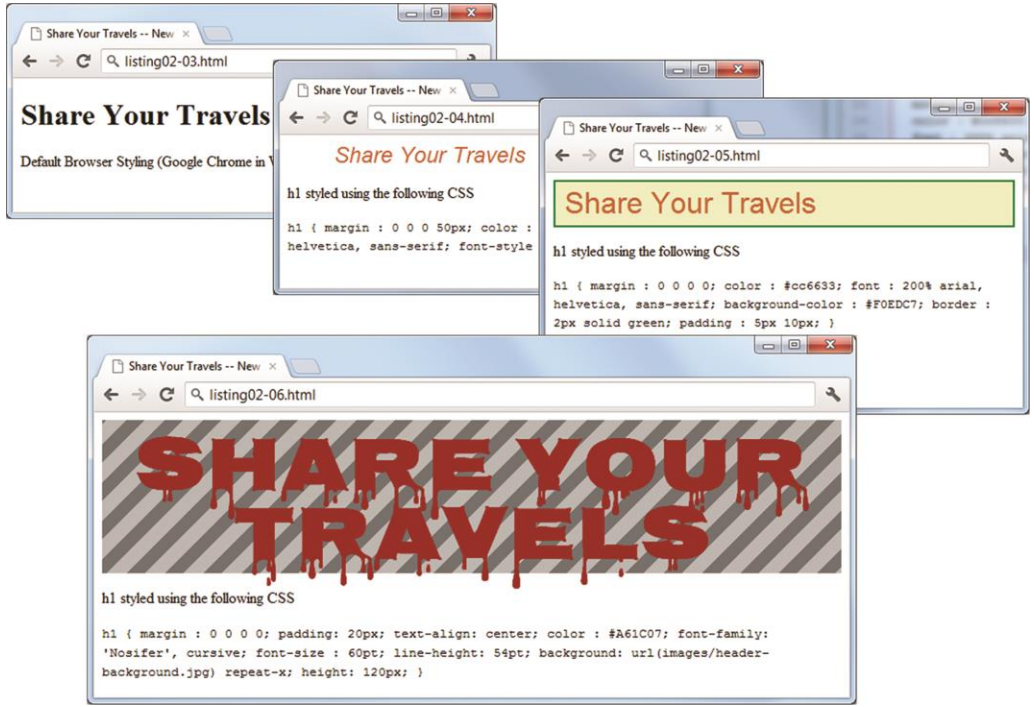
Headings

- HTML provides six levels of heading (h1 through h6)
- They are an essential way for document authors to show their readers the structure of the document
- Headings are also used by the browser to create a **document outline** for the page.
- Choose the heading level because it is appropriate semantically NOT because of its default presentation (e.g., choosing <h3> because you want your text to be bold and 16pt).

Heading Styles

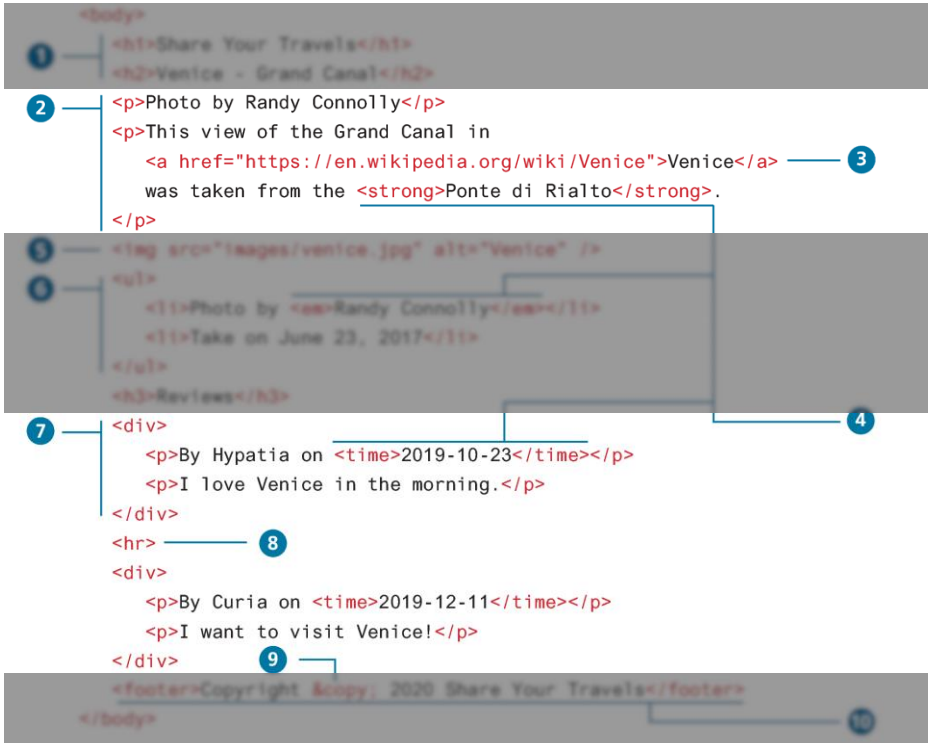
The browser has its own default styling for each heading level.

These are easily modified and customized via CSS (next chapter)



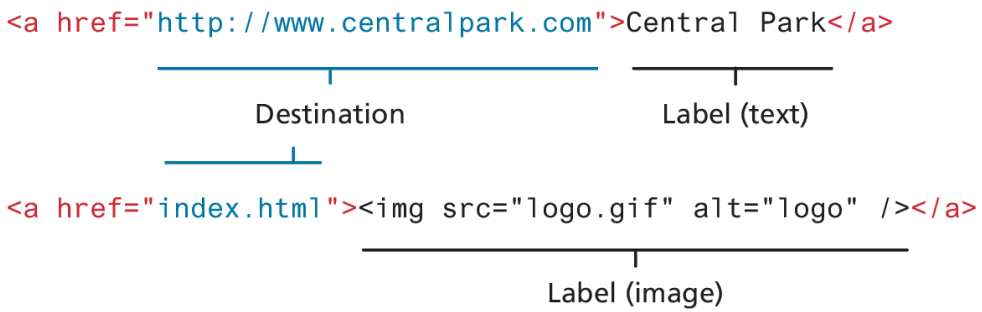
Paragraphs and Divisions

- The <p> tag is a container. It can contain HTML and other **inline HTML elements**
- is also a container element The element has no intrinsic presentation or semantic value;
- <hr> element is used to add a “break” between paragraphs or elements.



HyperLinks

- Links are created using the <a> element (the “a” stands for anchor).
- A link has two main parts: the **destination** and the **label**.



Kinds 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 (whether on the same site or on an external site).
- 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.
- Links that are instructions to the mobile browser to make a phone call.
- Links that are instructions to other programs (e.g., Skype, FaceTime, FaceBook Messenger).

Absolute and Relative URLs

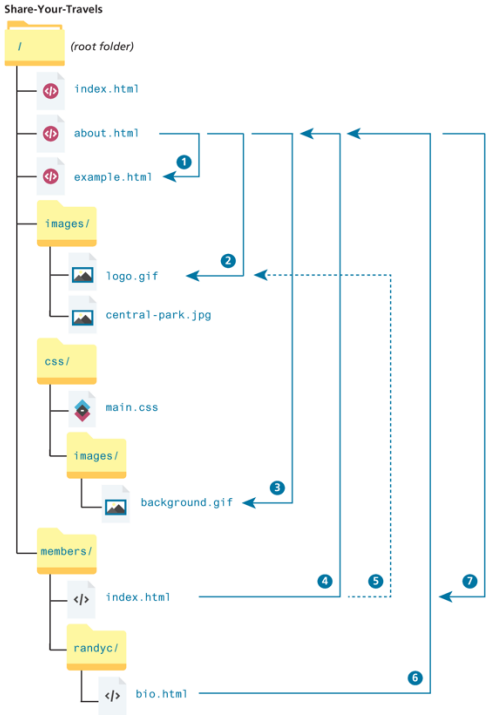
When referencing a page or resource on an **external site**, a full **absolute URL reference** is required

- Full URL with a protocol (typically, http:// or https://), the domain name, any paths, and the file name of the desired resource.

When referencing a resource that is on the **same server**, you can use **relative referencing**.

- If the URL does not include the "**http://**" then the browser will request the current server for the file.

Relative URLs



1. **Same Directory** To link to a file within the same folder, simply use the file name.
2. **Child Directory** To link to a file within a subdirectory, use the name of the subdirectory and a slash before the file name.
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.
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 "**../**".
5. **Sibling Directory** Use "**../**" to move up to the appropriate level, and then use the same technique as for child or grandchild directories.
6. **Root Reference** In this approach, begin the reference with the root reference (the "**/**"), and then use the same technique as for child or grandchild directories.

See Table 3.1 for examples (p. 97)

Inline Text Elements

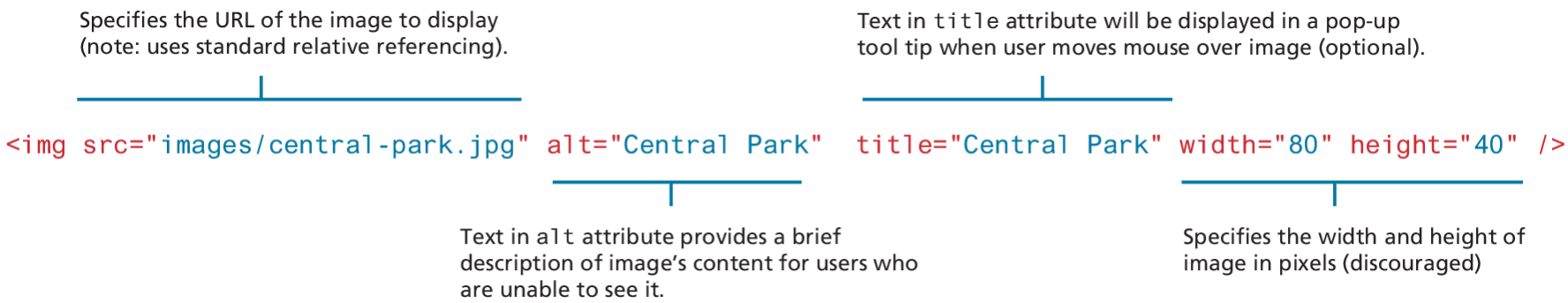
- inline elements because they do not disrupt the flow of text (i.e., cause a line break).
- HTML defines over 30 of these elements.
- Table 3.2 lists some of the most commonly used of these elements.

Common Text-Level Semantic Elements

- **<a>** Anchor used for hyperlinks.
- **<abbr>** An abbreviation
- **
** Line break
- **<cite>** Citation (i.e., a reference to another work)
- **<code>** Used for displaying code, such as markup or programming code
- **** Emphasis
- **<mark>** For displaying highlighted text
- **<small>** For displaying the fine-print, that is, “nonvital” text, such as copyright or legal notices
- **** The inline equivalent of the element. It is generally used to mark text that will receive special formatting using CSS
- **** For content that is strongly important
- **<time>** For displaying time and date data

Images

- Chapter 6 examines the different types of graphic file formats.
- Note the key attributes of the <img> element.
- Attributes such as title, width, and height are optional



Character Entities

Character entities are special characters for symbols for which there is either no easy way to type them via a keyboard 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

Entity Examples

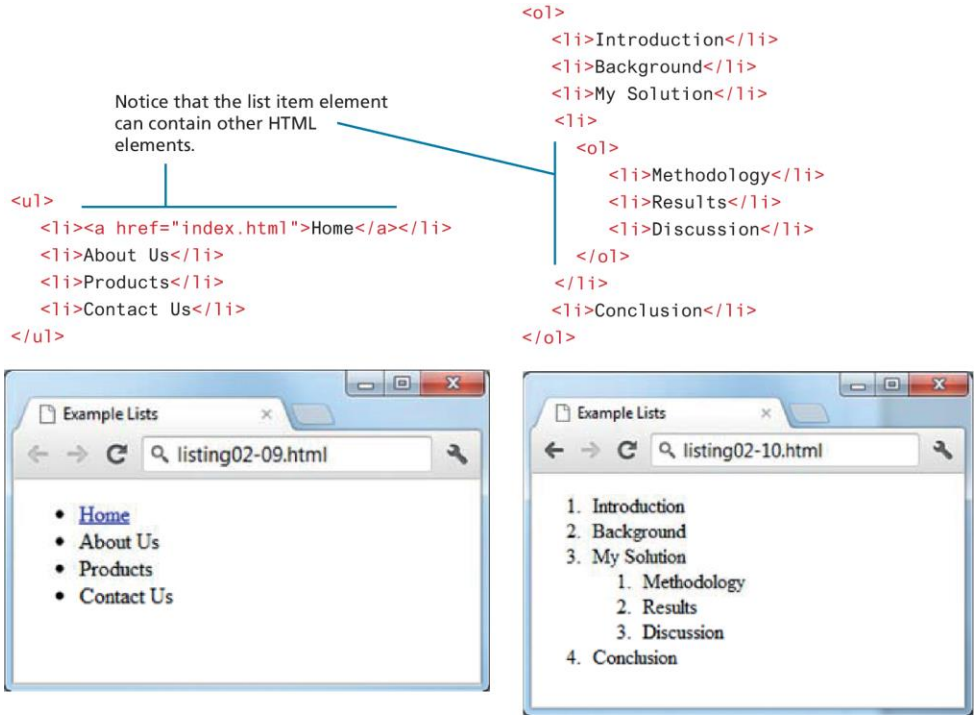
Entity Name	Entity Number	Description
** **		Nonbreakable space.
<	<	Less than symbol (“<”).
>	>	Greater than symbol (“>”).
©	©	The © copyright symbol
€	€	The € euro symbol.
™	™	The ™ trademark symbol.

Lists

- **Ordered lists** Collections of items that have a set order

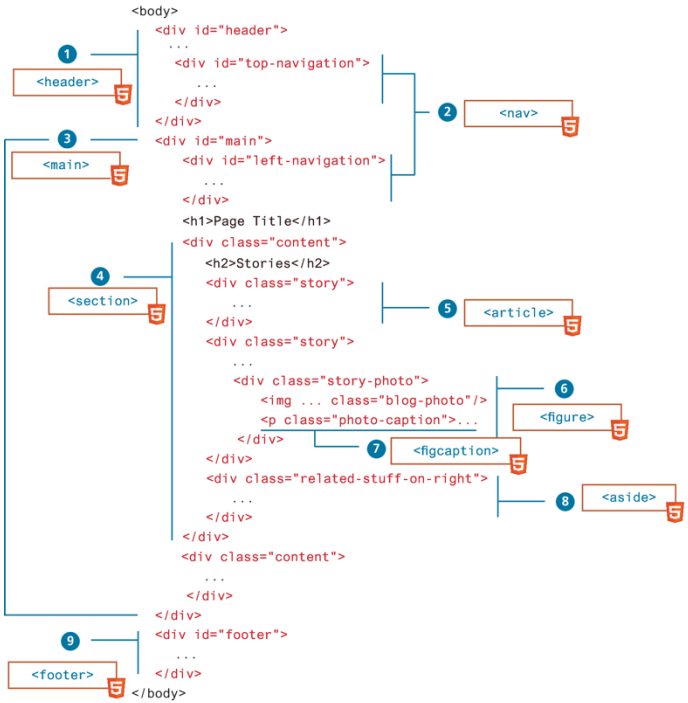
- **Unordered Lists**
  Collections of items in no particular order

- **Description Lists**
  Collection of name and description/definition pairs.
  <dl> <dt>



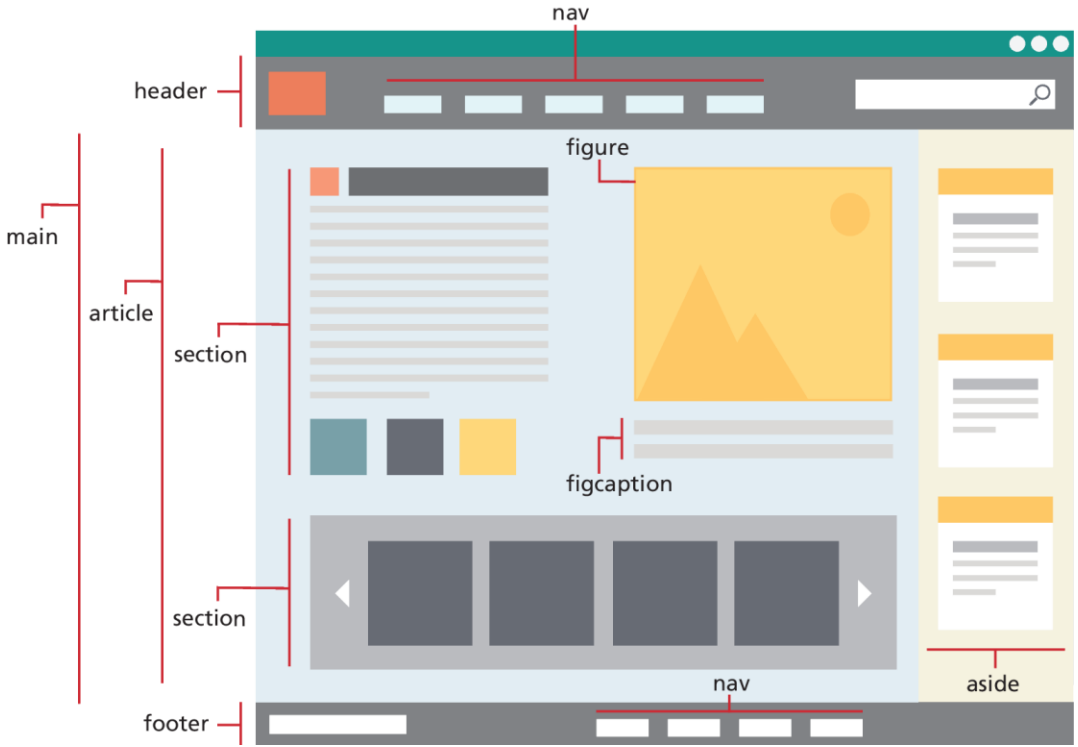
HTML5 Semantic Structure

- So far, the main semantic elements you have seen are headings, paragraphs, lists, some inline elements and the semantic block element, the division (i.e., element).
- HTML5 semantic elements allow to replace some of your sprawl with cleaner and more self-explanatory elements



HTML5 Semantic Structure Elements

- Header
- Nav
- Main
- Section
- Article
- Figure
- Figcaption
- Aside
- Footer



Using Semantic Elements

- HTML5 semantic elements do not apply any special presentation giving them great flexibility.
- Article and section, for instance, can be used many ways when designing your website.

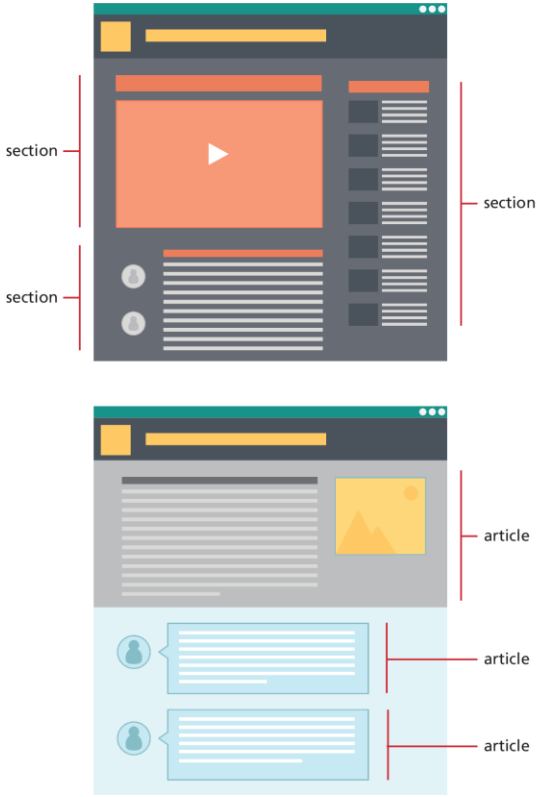


Figure and figcaption

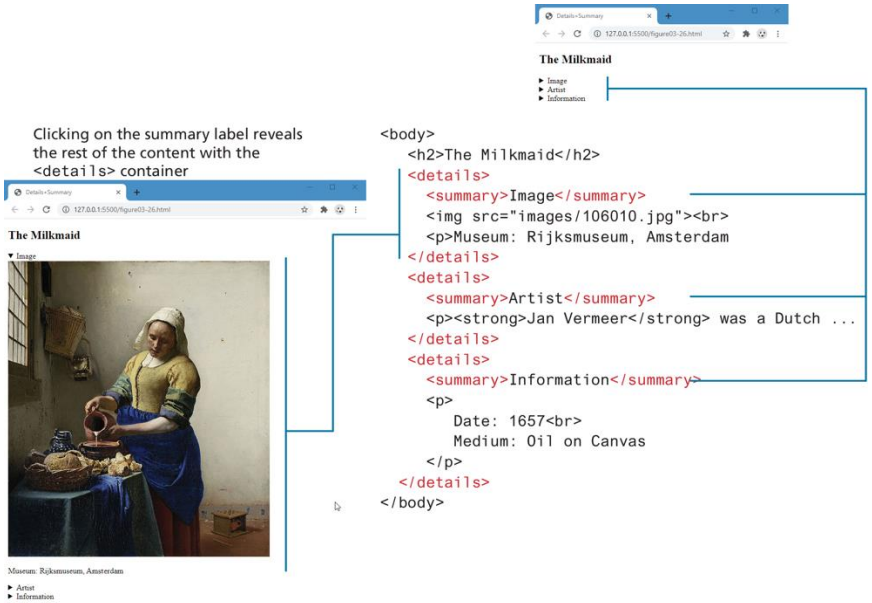
The <figure> element can be used not just for images but for any type of *essential* content that could be moved to a different location in the page or document, and the rest of the document would still make sense.

Figure 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>
```

Details and Summary



- The `<details>` and `<summary>` elements provide a way of semantically relating a summary and a details.
- For browsers that support these elements, *accordion functionality* is included as well (thus no JavaScript programming is required).

Additional Semantic Elements

- The `<blockquote>` element is a way to indicate a quotation from another source.
- The `<address>` element indicates that the enclosed HTML contains contact information for a person or organization.
- Additional list in Table 3.2

Key Terms

- absolute referencing
- accessibility
- ancestors
- body
- Cascading Style Sheets
- (CSS)
- character entity
- description lists
- descendants
- directory
- document outline
- Document Object Model
- empty element
- folder
- Head
- HTML attribute
- HTML element
- HTML validators
- inline HTML elements
- maintainability
- markup
- markup language
- ordered lists
- pathname
- performance
- polyfill
- quirks mode
- Recommendations
- relative referencing
- root element
- root reference
- schemas
- search engine optimization
- semantic HTML
- specifications
- standards mode
- tags
- unordered lists
- UTF-8
- WHATWG
- World Wide Web
- Consortium
- W3C
- XHTML 1.0
- XML

Copyright