‘[HTML 5’]

DEESHA

Vishal Shah | M. 98508 10100

DEESHA COMPUTER EDUCATION | VISHUPRASAD APARTMENT, NR. DANDEKAR SHOPPE, VISHRAMBAG, SANGLI. EMAIL : MAIL.DEESHA@GMAIL.COM | PHONE : +91 233 6600371

HTML

HTML stands for **HyperText Markup Language**. HTML is the main markup language for describing the structure of Web pages

Hypertext is text displayed on a computer or other electronic device with references to other text that the user can immediately access, usually by a mouse click or key press.

Apart from text, hypertext may contain tables, images and other presentational elements. It is an easy-to-use and flexible format to share information over the Internet.

Markup Languages use sets of **markup tags** to characterize text elements within a document, which gives instructions to the Web browser on how the document should appear

- HTML is the language for publishing web pages on the **WWW** (World-Wide Web, or World-Wide Wait?).
- HTML is a Document Description Language (aka Document Markup Language). HTML is NOT a programming language like C/C++/C#/Java, which is used to implement programming algorithm.
- An HTML document is a text document, and it is human-readable

HTML Gives You Ability To:

- Publish documents online with text, headings, images, lists, tables, etc.
- Accessing online information or other web resources such as images, videos or other HTML document via hyperlinks.
- Create forms to collect user input like login information, feedback, comments or conducting transactions with remote servers, etc.
- Include videos, sound clips, flash movies, applications and other HTML document directly inside an HTML document.

***Note:** HyperText Markup Language (HTML), as described earlier is a markup language not a programming language*

History of HTML

The first web browser was introduced in 1993 and the name was MOSAIC. The development of MOSAIC was at the **NCSA (National Center for Supercomputing Applications**). Later it was discontinued to development on 7th of January 1997. Still the people were using the nonstandard version of HTML.

The standard version came into existence in 1995, when HTML 2.0 was announced. Later after two years HTML 3.0 and after two years HTML 4.01 was announced. And still we are using the milestone of HTML 4.01.

HTML5 is the newest version of Hyper Text Markup Language. The First Draft of HTML5 Was announced in January 2008. And amazingly HTML5 has a broad browser support. Though the HTML5 is still under developing phase. And a lot of organizations are working and planning for the development of HTML5.

HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).

The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, and Google Gears.

New Features

HTML5 introduces several new elements and attributes that helps in building a modern website. Following are great features introduced in HTML5.

- **New Semantic Elements** – These are like <**header**>, <**footer**>, and <**section**>.
- **Forms 2.0** – Improvements to HTML web forms where new attributes have been introduced for <input> tag.
- **Persistent Local Storage** – To achieve without resorting to third-party plugins.
- **Web Socket** – A next-generation bidirectional communication technology for web applications.
- **Server-Sent Events** – HTML5 introduces events which flow from web server to the web browsers and they are called Server-Sent Events (SSE).
- **Canvas** – This supports a two-dimensional drawing surface that you can program with JavaScript.
- **Audio & Video** – You can embed audio or video on your web pages without resorting to third-party plugins.
- **Geolocation** – Now visitors can choose to share their physical location with your web application.
- **Microdata** – This lets you create your own vocabularies beyond HTML5 and extend your web pages with custom semantics.
- **Drag and drop** – Drag and drop the items from one location to another location on a the same webpage.

HTML Versions - HTML4.01, XHTML1.0 and HTML5

Today, the W3C (World-Wide Web Consortium) (http://www.w3c.org) maintains the specifications of HTML and CSS (and many other related web technologies).

HTML has gone through these changes:

- **HTML Draft** (October1991): Tim Bernes-Lee (of CERN) proposed the early HTML for sharing of document in a hypertext system
- **HTML 2.0** (November 1995): Published as IETF RFC 1866
- **HTML 3.2** (January 1997): Published as W3C Recommendation
- **HTML 4.0** (December 1997): Published as W3C Recommendation, with strict, transitional and frameset
- **HTML 4.01** (December 1999): Touched up HTML 4.0. The supposedly *final* HTML specification published by W3
- **XHTML 1.0** (January 2000): W3C considered HTML 4.01 as the final release for HTML, and moved on to develop XHTML 1.0 with stricter rules and syntaxes, followed by XHTML 2.0. XHTML 2.0, although theoretically elegant, is impractical as it is not backward compatible with HTML4/XHTML1.0. A rebel group called WHATWG (Web

Hypertext Application Technology Working Group) continued to work on extending HTML with more features in a backward-compatible manner. In 2004, WHATWG released HTML5. By 2007, HTML5 has captured the attention of the developers. W3C decided to abandon the XHTML 2.0 and embraced the HTML5

- **HTML 5** (October 2014): Published as W3C Recommendation

Limitations of HTML 4

- The previous version of HTML 4.01 is released in 1999 and after that Internet has changed a lot.
- DOCTYPE is much longer as HTML4 is based on SGML.
- Audio and Video are not part of HTML4 specification
- Vector Graphics is possible with the help of technologies such as VML, Silverlight, Flash etc.
- It is almost impossible to get true Geo Location of user browsing any website especially if it comes to mobile devices.
- Browser cache can be used as temporary storage.
- JavaScript runs in same thread as browser interface.

Advantages of HTML 5

- HTML5 is a cooperation between the World Wide Web Consortium (**W3C**) and the Web Hypertext Application Technology Working Group (**WHATWG**).
- All the New features of HTML5 are based on **HTML, CSS, DOM, and JavaScript**. HTML5 is designed to replace HTML 4, XHTML and HTML DOM Level 2.
- It is designed to deliver rich content without the need for additional plugins which was explicitly done in previous versions of HTML.
- The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, and Google Gears.
- HTML5 is still a work in progress. However, the major browsers support many of the new HTML5 elements and APIs.
- HTML5 is device independent.
- HTML5 has more markup to replace scripting.

Markup Tags

HTML uses markup tags, such as <p> (for Paragraph), <h1> to <h6> (for Heading Level 1 to 6), <img> (for Image), <a> (for Anchor or Hyperlink), to markup a document. HTML markup tags perform these functions:

- Layout the documents, e.g., `<p>` (layout as a paragraph), `<h1>` to `<h6>` (layout as heading level 1 to 6), `<br>` (perform a line break), `<hr>` (draw a horizontal rule), `<table>` (tabulating data), `<ol>` (layout an ordered list).

- Provide link (called *hyperlink*) to another HTML document, via the `<a>` (Anchor tag). These hyperlinks, a distinct feature in HTML, greatly help the users in navigating the web and enrich the users' experience. Hyperlinks make the HTML popular.

- Embed images, audios, videos, programs (in JavaScript, VBScript, Applet, Flash, or MS ActiveX control), and objects within an HTML document. HTML is *multimedia*! The hypertext document may contain texts, images, audios, videos, and even programs.

Separating Content and Presentation

The purpose of a markup language is to relieve the content provider from worrying about the actual appearance of the document. The author merely indicates (via markup tags) the *semantic meaning* of the words and sentences (such as paragraph, heading, emphasis, and strong), and leave it to the browser to interpret the markups and render the document for display on the screen. In other words, it allows the *separation of content and presentation*. The content provider focuses on the document contents, while the graphic designer concentrates on the view and presentation.

Nowadays, HTML should be used solely to markup the contents, while its companion technology known as CSS (Cascading Style Sheet) be used for defining the presentation of the document, so as to *separate the content and presentation*.

These are the common pitfalls in older HTML documents and you should avoid:

- Do not specify "appearance" properties, such as foreground and background color, text alignment, font face, font size, border, margin and padding, in the HTML document. Use an external CSS instead to set the appearance and presentation. Presentation-related tags (such as `<font>`, `<center>`) and attributes (such as `align`, `bgcolor`, `link`, `vlink`, `alink`) have been deprecated in HTML 4, in favor of CSS.

- Do not use nested tables or frame for formatting the document, use `<div>` and `<span>`, or HTML5 new tags such as `<header>`, `<footer>`, and `<section>`.

HTML Tags and Elements

HTML is written in the form of HTML elements consisting of markup tags. These markup tags are the fundamental characteristic of HTML. Every markup tag is composed of a keyword, surrounded by angle brackets, like `<html>`, `<head>`, `<body>`, etc.

HTML tags normally come in pairs like `<html>` and `</html>`. The first tag in a pair is the opening tags the second tag is the closing tags. An opening tag and a closing tag are identical, except a slash (/) after the opening angle bracket of a closing tag, to tell the browser that the command has been completed. In between these tags you can add headings, paragraphs of text, tables, forms, images, videos etc. For example, a paragraph, which is represented by the `<p>` element, would be written as:

```
<p> This is a paragraph. </p>
```

HTML Document

All HTML document seem to be plain text files. They contain no images, sounds, videos etc. but just plain text. However, they may contain links to images, sounds and videos.

> ***Note:*** ***The web browsers does not display the HTML tags, but uses the tags to interpret the content of the web pages***

An HTML file is simply a text file saved with an .html or .htm extension (i.e. as opposed to a .txt extension)

First HTML Document

Step 1: Creating the HTML file

Open up your computer's plain text editor and create a new file

> ***Tip:*** ***We suggest you to use only the simplest text editors such as Notepad (under Windows), TextEdit (on the Mac). Once you understand the basic principles, you may switch to more advanced tools such as Dreamweaver.***

Step 2: Type some HTML code

Start with an empty window and type the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>A simple HTML document</title>
</head>
```

```
<body>
    <p>Hello World!<p>
</body>
</html>
```

- DOCTYPE declaration — A line defines the HTML5 document type.
- A declarative header section (enclosed by the HEAD element) — Provides information about the document, including its title, style information, and scripts.
- A document body (enclosed by the BODY element) — Contains the document's actual content that is rendered in the web browser and displayed to the user.

Step 3: Saving the file

Now save the file on your desktop as "myfirstpage.html ".

> ***Note:*** ***It is important that the extension .html is specified — some text editors, such as Notepad, will automatically save it as .txt otherwise.***

To open the file in a browser. Navigate to your file then double click on it. It will open in your default Web browser. (If it does not, open your browser and drag the file to it.)

Example: Basic Layout of an HTML Document

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Basic HTML Document Layout</title>
</head>
<body>
  <h1>My First Web Page</h1>
  <hr>
  <p>This is my <strong>first</strong> web page written in HTML.</p>
  <h3>HTML</h3>
  <p>HTML uses <em>markup tag</em> to <em>markup</em> a document.</p>
</body>
</html>
```

How it Works?

1. An HTML document begins with a Document-Type declaration `<!DOCTYPE html>` (Line 1) to identify itself as an HTML document to the browser.
2. The HTML content is contained within a pair of `<html>...</html>` tags. You can specify the *default* language of your document via *attribute* `lang="en"` (for English) in the `<html>` opening tag.

3. There are two sections in the document: `HEAD` and `BODY`, marked by `<head>...</head>` and `<body>...</body>` tags, respectively.
4. In the `HEAD` section:
   a. The `<meta charset="utf-8">` element (Line 4) specifies the *character encoding scheme* of the document. Today, virtually all (English) HTML documents are encoded using the `UTF-8` character encoding scheme, which is compatible with ASCII code for English alphabets and allow you to include other Unicode characters (such as Chinese, Japanese and Korean) efficiently.
   When saving your file, you need to choose "UTF-8 encoding" in the "save-as" dialog menu.
   b. The `<title>...</title>` element (Line 5) provides a *descriptive title* to the page. The browser displays the title on the title-bar of the tab/window.
5. In the BODY section:
   a. The <h1>...</h1> *container* tags (Line 8) mark the enclosing texts as Level-1 Heading. There are six levels of heading in HTML, from <h1>...</h1> (largest) to <h6>...</h6>. Line 11 uses a <h3>...</h3> (Heading Level-3).
   b. The <hr> *standalone* element (Line 9), which does not enclose text content, draws a horizontal rule (or line).
   c. The <p>...</p> container tags (Line 10 and 12) mark the enclosing texts as a paragraph. <p>...</p> is the most frequently-used tag in HTML.
   d. The <strong>...<strong> tags (nested under the <p>...</p> in Line 10) specify "strong emphasis" for its content - rendered in bold by the browser. Similarly, the nested <em>...</em> tags (Line 12) specify "emphasis" - rendered in italic by the browser

View Page Source

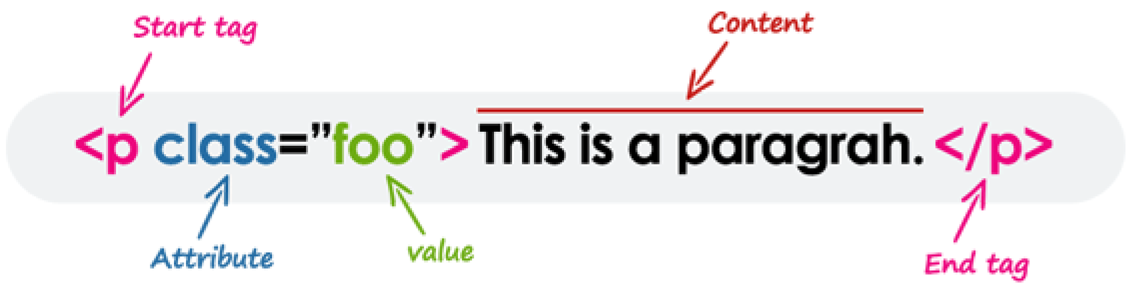
You can read the HTML source code by right-clicking on the page and select "View Source" (or "View Page Source", or "Show Page Source"). Try it out.

Note: For Mac's Safari, you may need to enable "Show Page Source" via "Preferences" ⇒ Advanced ⇒ "Show Develop menu in menu bar".

HTML Element Syntax

HTML Elements represent semantics, or meaning. For example, The `title` element represents the title of the document. Most HTML elements are written with a *start tag* (or opening tag) and an *end tag* (or closing tag), with the content in between.

Elements can also contain attributes that define additional properties of an element. For example, a paragraph, which is represented by the `p` element, would be written as:



Note: *All elements don't require the end tag to be present. These are referred as empty elements, self-closing elements or void elements. Learn more.*

HTML Tags Vs Elements

Technically, an HTML element is the collection of start tag, its attributes, an end tag and everything in between. On the other hand, an HTML tag either opening or closing is used to mark the start or end of an element.

However, in common usage the terms HTML element and HTML tag are interchangeable i.e. a tag is an element is a tag. For simplicity's sake of this website, the terms "tag" and "element" are used to mean the same thing — as it will define something on your web page

Case Insensitivity in HTML Tags and Attributes

In HTML, tag and attribute names are not case-sensitive. It means the tag `<P>`, and the tag `<p>` defines the same thing in HTML which is a paragraph.

But in `XHTML` they are case-sensitive and the tag `<P>` is different from the tag `<p>`.

Tip: *We recommend using lowercase for tag and attributing names in HTML, since by doing this you can make your document more compliant for future upgrades.*

Empty HTML Elements

Empty elements (also called **self-closing** or **void elements**) are not container tags — that means, you can not write `<hr>some content</hr>` or `<br>some content</br>`.

A typical example of an empty element, is the `<br>` element, which represents a line break

```
<p>This paragraph contains <br> a line break. </p>
```

> ***Note:*** ***In HTML, a self-closing element is written simply as <br>. In XHTML, a self-closing element requires a space and a trailing slash, such as <br />.***

Nested HTML Elements

Most HTML elements can contain any number of further elements, which are, in turn, made up of tags, attributes, and content or other elements.

The following example shows two elements: `<em>` element nested inside `<p>` element.

```
<p class="example">Here is some <em>emphasized</em> text.</p>
```

HTML Elements Types

Elements can be placed in two distinct groups: **block level** and **inline level** elements. The former make up the document's structure, while the latter dress up the contents of a block

Block-level Elements and Block Boxes

Block-level elements are those elements that are formatted visually as blocks (i.e. takes up the full width available), with a line break before and after the element. For example, paragraph element (`<p>`), headings (`<h1> to <h6>`), divisions (`<div>`) etc.

Generally, block-level elements may contain inline elements and other block-level elements

Inline-level Elements and Inline Boxes

Inline-level elements are those elements of the source document that do not form new blocks of content; the content is distributed in lines. For example, emphasized pieces of text within a paragraph (`<em>`), spans (`<span>`), strong element (`<strong>`) etc.

Inline elements typically may only contain text and other inline elements

> ***Note:*** ***Unlike block-level elements, an inline-level element only takes up as much width as necessary, and does not force line breaks.***

HTML Attributes

Certain parameters are frequently included within the opening tag to provide additional element properties (such as colorization, measurement, location, alignment, or other appearances) to the data between the markup tags. These parameters are called **Attributes**.

Note: ***Attributes are always specified in the start tag (or opening tag), and they can only contain the value of the attribute.***

Attribute Values

Most attributes require a value. In HTML, the value can be left unquoted if it doesn't include spaces (`name=value`), or it can be quoted with single or double quotes (`name='value'` or `name="value"`). By the way, it is recommended to enclose Attribute values in quotes.

For example, the `<abbr>` element, which represents an abbreviation, expects a `title` attribute with its expansion. This would be written as:

```
<abbr title="Hyper Text Markup Language">HTML</abbr>
```

Tip: ***In some rare situations, when the attribute value itself contains quotes, it is necessary to wrap attribute value inside single quotes: name='John "Williams" Jr.'***

Attribute Example

In the example below href is attribute and the link provided is its value.

HTML links are defined with the `<a>` tag. You will learn about links in upcoming sessions

```
!-- link to microsoft.com -->
<a href="http://www.microsoft.com">Microsoft</a>
<!-- link to google.com -->
<a href="http://www.google.com">Google</a>
```

Note: ***Attribute values are generally case-insensitive, except certain attribute values, like the id and class attributes. However, World Wide Web Consortium (W3C) recommends lowercase for attributes values in their specification.***

HTML Headings

Headings help in defining the hierarchy and the structure of the web page.

HTML uses six levels of heading tags <h1> to <h6>; the higher the heading level number, the greater it's importance — so <h1> defines the most important heading, whereas the <h6> defines the least important heading in the document.

```
<h1>This is a heading 1</h1>
<h2>This is a heading 2</h2>
<h3>This is a heading 3</h3>
<h4>This is a heading 4</h4>
<h5>This is a heading 5</h5>
<h6>This is a heading 6</h6>
```

Note: ***Each time you place a heading tag, your web browser built-in style sheets automatically create some empty space before and after each heading. Use CSS margin property to override browser's default style sheet.***

Importance of Headings

- HTML headings provide valuable information by highlighting important topics and the structure of the document.
- Don't use headings to make text BIG or bold. Use it only for highlighting the heading of your document and to show the document structure.
- As search engines use headings to index the structure and content of your web pages so use it very wisely in your webpage.
- Use <h1> headings as main headings, followed by <h2> headings, then the less important <h3> headings, and so on.

Tip: ***A document generally should have exactly one <h1> element to mark the most important heading, followed by the <h2>, <h3> and so on.***

HTML Paragraphs

Paragraph element used to publish text on the web pages.

Paragraphs are defined with the <p> tag. Paragraph tag is very basic and typically the first tag you will need to publish your text on the web pages.

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Note: ***Browsers built-in style sheets automatically create some space above and below the content of a paragraph, but you can override it using CSS.***

Closing a Paragraph Element

In HTML 4 and earlier versions, it was enough to initiate a new paragraph using the opening tag. Most browsers will display HTML correctly even if you forget the end tag. For example:

```
<p>This is a paragraph.
<p>This is another paragraph.
```

The example above will work in most of the browsers, but don't rely on it. Forgetting the end tag can produce unexpected results or errors.

Note: ***For the purposes of forwards-compatibility and good coding practice, it's advisable to use both the opening and closing tags for the paragraphs.***

HTML Line Breaks

The `<br>` element is used to insert a line break without starting a new paragraph.

```
<p>This is a paragraph <br> with line break.</p>
<p>This is <br>another paragraph <br> with line breaks.</p>
```

Note: ***Don't use empty paragraph i.e. <p></p> to add extra space in your web pages. The browser may ignore the empty paragraph since it is logical tag. Use the CSS margin property instead to adjust the spaces.***

HTML Comments

Comments are usually added with the purpose of making the source code easier to understand. It may help other developer (or you in the future when you edit the source code) to understand what you were trying to do with the HTML. Comments are significant to programmers but typically ignored by browsers.

An HTML comment begins with `<!--`, and ends with `-->`, See the example below:

```
<-- This is an HTML comment -->
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

HTML Spaces

Normally the browser will display the multiple spaces created inside the HTML code by pressing the *space-bar key* on the keyboard as one space, while multiple line breaks created through pressing the enter key also displayed as single space. Insert `&nbsp;` for creating extra spaces, and `<br>` tag to create line breaks inside your HTML document.

```
<p>multiple&nbsp;&nbsp;&nbsp;spaces.</p>
<p>multiple<br><br>line<br><br><br>breaks.</p>
```

New Semantic Elements in HTML5

Many web sites contain HTML code like: <div id="nav"> <div class="header"> <div id="footer"> to indicate navigation, header, and footer.

HTML5 offers new **semantic** elements to define different parts of a web page. The following tags have been introduced for better structure

If you carefully plan the structure of your HTML documents, you can help computers make sense of the meaning of your content. Proper syntax is important for sure, but it basically just provides parsers, search engines, and assistive technologies with a meaningless bunch of data.

Non-Semantic Elements

Non-semantic elements don't have any special meaning, the heretical relationships between them are merely illusory. The most widely used examples of non-semantic HTML tags are the **`<div></div>`** and the **<span></span>** tags.

Document Outline in HTML5

The document outline is the structure of an HTML document. It shows how elements are related to each other. The document outline has been generated solely by mapping elements, such as headings, table titles, form titles, and others in the earlier versions of HTML such as HTML4.01 and XHTML.

- **`<section></section>`** for sections grouped around a specific theme
- **`<article></article>`** for complete or self-contained compositions such as a blog post or a widget
- **`<nav></nav>`** for navigation blocks
- **`<aside></aside>`** for complementary content such as sidebars

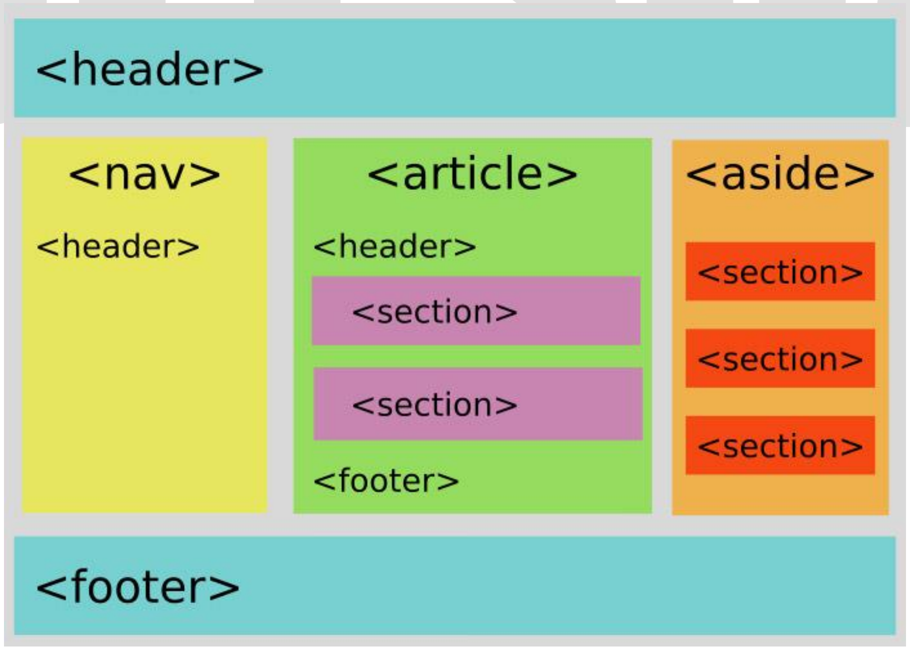
Tips for Semantically Structured Content

If we want to create a well-structured document outline, we need to pay attention to the following rules:

1. The outermost sectioning element is always the **<body></body>** tag.
2. Sections in HTML5 can be nested.
3. Each section has its own heading hierarchy. Each of them (even the innermost nested section) can have an **h1** tag.
4. While the document outline is primarily defined by the 5 sectioning elements, it also needs proper headings for each section.
5. It's always the first heading element (let it be h1 or a lower rank heading tag) that defines the heading of the given section. The following heading tags inside the same section need to be relative to this. (If the first heading is an h3 inside a sectioning element, don't put an h3 after that.)
6. The sections defined by the **<nav></nav>,** and the **<aside></aside>** tags don't belong to the main outline of the HTML document, they are usually not rendered initially by assistive technologies.
7. Each section (body, section, article, aside, nav) can have their own **<header></header>** and **<footer></footer>** tags, that defines the header (such as logo, author's name, dates, meta info, etc.) and the footer (copyright, notes, links, etc.) of that section

Example: A Semantic Outline

Let's see an example for a semantic document outline to see clearer how it works. Our example code will result in the following document structure:



```
<body>

  <header>
    <h1>Welcome On Our Website!</h1>
    <p>Here is our logo and slogan.</p>
  </header>

  <nav>
    <header>
      <h2>Choose Your Interest</h2>
    </header>
    <ul>
      <li>Menu 1</li>
      <li>Menu 2</li>
      <li>Menu 3</li>
    </ul>
  </nav>

  <article>
    <header>
      <h1>Title of Article</h1>
      <h2>Subtitle of Article</h2>
    </header>

    <section>
      <h3>First Logical Part (e.g. "Theory")</h3>
      <p>Paragraph 1 in first section</p>

      <h4>Some Other Subheading in First Section</h4>
      <p>Paragraph 2 in first section</p>
    </section>

    <section>
      <h3>Second Logical Part (e.g. "Practice")</h3>
      <p>Paragraph 1 in second section</p>
      <p>Paragraph 2 in second section</p>
    </section>

    <footer>
      <h4>Author Bio</h4>
      <p>Paragraph in Article's Footer</p>
    </footer>

  </article>

  <aside>

    <h2>Get To Know Us Better</h2>
```

```
        <section>
          <h3>Popular Posts</h3>
          <ul>...</ul>
        </section>

        <section>
          <h3>Partners</h3>
          <ul>...</ul>
        </section>

        <section>
          <h3>Testimonials</h3>
          <ul>...</ul>
        </section>

      </aside>

      <footer>
        <ul>
          <li>Copyright</li>
          <li>Social Media Links</li>
        </ul>
      </footer>

    </body>
```

Attributes

The elements may contain attributes that are used to set various properties of an element.

Some attributes are defined globally and can be used on any element, while others are defined for specific elements only. All attributes have a name and a value and look like as shown below in the example.

Following is the example of an HTML5 attributes which illustrates how to mark up a div element with an attribute named class using a value of "example" –

```
<div class="example">...</div>
```

Attributes may only be specified within start tags and must never be used in end tags.

HTML5 attributes are case insensitive and may be written in all uppercase or mixed case, although the most common convention is to stick with lowercase

Standard Attributes

The attributes listed below are supported by almost all the HTML 5 tags.

Attribute	Options	Function
accesskey	User Defined	Specifies a keyboard shortcut to access an element.
align	right, left, center	Horizontally aligns tags
background	URL	Places an background image behind an element
bgcolor	numeric, hexidecimal, RGB values	Places a background color behind an element
class	User Defined	Classifies an element for use with Cascading Style Sheets.
contenteditable	true, false	Specifies if the user can edit the element's content or not.
contextmenu	Menu id	Specifies the context menu for an element.
data-XXXX	User Defined	Custom attributes. Authors of a HTML document can define their own attributes. Must start with "data-".
draggable	true,false, auto	Specifies whether or not a user is allowed to drag an element.
height	Numeric Value	Specifies the height of tables, images, or table cells.
hidden	hidden	Specifies whether element should be visible or not.
id	User Defined	Names an element for use with Cascading Style Sheets.
item	List of elements	Used to group elements.
itemprop	List of items	Used to group items.
spellcheck	true, false	Specifies if the element must have it's spelling or grammar checked.
style	CSS Style sheet	Specifies an inline style for an element.
subject	User define id	Specifies the element's corresponding item.
tabindex	Tab number	Specifies the tab order of an element.
title	User Defined	"Pop-up" title for your elements.
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
width	Numeric Value	Specifies the width of tables, images, or table cells.

HTML Text Formatting

Using the text formatting tags you can make some text on your web pages to appear differently than normal text content.

HTML Text Formatting Tags

Tag	Description
<b>	Defines bold text.
<del>	Defines deleted text.
<em>	Defines emphasized text.
<i>	Defines italic text.
<ins>	Defines inserted text.
<mark>	Defines marked/highlighted text.
<small>	Defines small text.
<strong>	Defines strong text.
<sub>	Defines subscripted text.
<sup>	Defines superscripted text.

HTML Citations, Quotations, and Definition Tags

Tag	Description
<abbr>	Defines an abbreviation.
<address>	Defines contact information for the author/owner of a document.
<bdo>	Defines the text direction.
<blockquote>	Defines a long quotation.
<q>	Defines a short quotation.
<cite>	Defines a citation.
<dfn>	Defines a definition term.

HTML Computer Output Tags

Tag	Description
<code>	Defines computer code text.
<kbd>	Defines keyboard text.
<samp>	Defines sample computer code.
<var>	Defines a variable.
<pre>	Defines preformatted text.

HTML Lists

HTML lists are used to group related pieces of information together.

Understanding HTML Lists

HTML lists are used to present list of information in well-formed and semantic way. There are three different types of list in HTML and each one has a specific purpose and meaning:

- Unordered list — Used to group a set of related items, in no particular order.
- Ordered list — Used to group a set of related items, in a specific order.
- Description list — Used to display a list of terms and their descriptions.

Note: Inside a list item you can put text, line breaks, images, links, etc. You can also place an entire list inside a list item to create nested list.

HTML Unordered Lists

An unordered list created using the `<ul>` tag, and each list item starts with the `<li>` tag. The list items in unordered lists are marked with bullets (small black circles), by default.

```
<ul>
    <li>Chocolate Cake</li>
    <li>Black Forest Cake</li>
    <li>Pineapple Cake</li>
</ul>
```

HTML Ordered Lists

An ordered list, created using the `<ol>` tag, and each list item starts with the `<li>` tag. Ordered list contain information where order should be emphasized.

The list items in ordered lists are marked with numbers.

```
<ol>
    <li>Mix ingredients</li>
    <li>Bake in oven for an hour</li>
    <li>Allow to stand for ten minutes</li>
</ol>
```

HTML Definition Lists

A definition list is a list of items, with a description of each item. The definition list created using `<dl>` tag. The `<dl>` tag is used in conjunction with `<dt>` — defines the item in the list, and `<dd>` describes the item in the list:

```
<dl>
    <dt>Bread</dt>
    <dd>A baked food made of flour.</dd>
    <dt>Coffee</dt>
    <dd>A drink made from roasted coffee beans.</dd>
</dl>
```

HTML Images

Images improve the appearance and illustrate many concepts of the web pages.

Inserting Images in HTML Documents

The web is not just about text, its multi-media and HTML's multimedia features allow you to include images, audio clips, video clips, and other multimedia element in the web pages.

The `<img>` tag is used to insert images in HTML documents. It is an empty element and contains attributes only. The syntax of `<img>` tag can be given with:

```
<img src="url" alt="some_text">

<img src="kites.jpg" alt="Flying Kites">
<img src="sky.jpg" alt="Cloudy Sky">
<img src="balloons.jpg" alt="Hot Air Balloons">
```

The Src Attribute of Images

Every image has a `src` attribute (*src* stands for *source*). The src attribute tells the browser where to find the image you want to display. The URL of the image provided as the value of src attribute points to the location where the image is stored.

For Example, An image named "sky.jpg", located in the "images" directory on "www.tutorialrepublic.com" has the URL: http://www.tutorialrepublic.com/images/sky.jpg

```
<img url="http://www.tutorialrepublic.com/images/sky.jpg" alt="Sky">
```

The Alt Attribute of Images

The `alt` attribute is the alternative description for an image, if the image cannot be displayed. The value of the alt attribute is an author-defined text.

```
<img url="http://www.tutorialrepublic.com/images/kites.jpg" alt="Kites">
```

Note: The required alt attribute provides alternative information for an image if a user for some reason cannot able to view it because of slow connection, an error in the src attribute, or if the user uses a screen reader.

Setting Width and Height of an Image

The width and height attributes are used to specify the height and width of an image.

The attribute values are specified in pixels by default

```
<img src="kites.jpg" alt="Flying Kites" width="300" height="300">
<img src="sky.jpg" alt="Cloudy Sky" width="250" height="150">
<img src="balloons.jpg" alt="Hot Air Balloons" width="200" height="200">
```

Note: It's a good practice to specify both the width and height attributes for an image. In the absence of these attributes the image loads with its original size that may cause distortion or flicker in your website layout.

Links (Hyperlinks or Web links)

A link or hyperlink is a connection from one web resource to another.

Links allow users to move seamlessly from one page to another, on any server anywhere in the world. A link has two ends called anchors and a direction. The link starts at the source anchor and points to the destination anchor, which may be any web resource (e.g. an image, an audio or video clip, an HTML document or an element within the document itself, etc.).

By default, links will appear as follows in most of the browsers:

- An unvisited link is underlined and blue.
- A visited link is underlined and purple.
- An active link is underlined and red.

HTML Link Syntax

Links are specified in HTML using the `<a>` tag. A link or hyperlink could be a word, group of words, or image.

```
<a href="url">Text link</a>
```

The href attribute in source anchor specifies the address of the destination anchor.

```
<a href="http://www.google.com/">Google Search</a>
<a href="kites.jpg"><img src="kites-thumb.jpg" alt="kites"></a>
```

Text Links

A webpage can contain various links that take you directly to other pages and even specific parts of a given page. These links are known as hyperlinks.

Hyperlinks allow visitors to navigate between Web sites by clicking on words, phrases, and images. Thus, you can create hyperlinks using text or images available on a webpage.

```
<a href="Document URL" ... attributes-list>Link Text</a>
```

The Target Attribute of Links

Target attribute tells the browser where to open linked document. There are four defined targets. Each target starts with an underscore (_): _blank, _parent, _self, _top.

```
<a href="http://www.tutorialrepublic.com/" target="_top">Tutorial Republic</a>
<a href="sky.jpg" target="_self"><img src="sky-thumb.jpg" alt="Cloudy Sky"></a>
<a href="http://www.google.com/" target="_blank">Google</a>
```

Option	Description
_blank	Opens the linked document in a new window or tab.
_self	Opens the linked document in the same frame.
_parent	Opens the linked document in the parent frame.
_top	Opens the linked document in the full body of the window.
targetframe	Opens the linked document in a named targetframe.

Use of Base Path

When you link HTML, documents related to the same website, it is not required to give a complete URL for every link. You can get rid of it if you use **<base>** tag in your HTML document header. This tag is used to give a base path for all the links. So, your browser will concatenate given relative path to this base path and will make a complete URL.

Example

Following example makes use of <base> tag to specify base URL and later we can use relative path to all the links instead of giving complete URL for every link.

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
<base href="http://www.deeshanotes.com/">
</head>
<body>
<p>Click following link</p>
<a href="/aboutus.aspx" target="_blank">About Us</a>
</body>
</html>
```

Linking to a Page Section

You can create a link to a particular section of a given webpage by using **name** attribute. This is a two step process.

First create a link to the place where you want to reach with-in a webpage and name it using <a...> tag as follows:

```
<h1>HTML Text Links <a name="top"></a></h1>
```

Second step is to create a hyperlink to link the document and place where you want to reach:

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

Setting Link Colors

You can set colors of your links, active links and visited links using `link`, `alink` and `vlink` attributes of <body> tag.

```
<body alink="#54A250" link="#040404" vlink="#F40633">
```

Image Links

We can use images for hyperlinks. It's simple to use an image as hyperlink. We just need to use an image inside hyperlink at the place of text as shown below:

```
<!DOCTYPE html>
<html>
<head>
<title>Image Hyperlink Example</title>
</head>
<body>
<p>Click following link</p>
<a href="http://www.google.com.com" target="_self">
   <img src="/images/logo.png" alt="Sample Tooltop text" border="0"/>
```

```
</a>
</body>
</html>
```

Creating Download Links

Placing files available for download is done in exactly the same fashion as placing text links, just point the destination URL to the file you want to be available for download.

In this case it is a Zip file called test.zip.

```
<a href="downloads/test.zip">Download Zip file</a>
<a href="downloads/masters.pdf">Download PDF file</a>
<a href="downloads/sample.jpg">Download Image file</a>
```

Note: Clicking a link that points to a PDF file or Image file will not cause it to download to your hard drive directly. It will only open the file in your web browser. Further you can save it to your hard drive.

Forms

An HTML form is a section of document that contains interactive controls that enable a user to submit information to a web server.

What is HTML Form

HTML Forms are required to collect different kinds of user inputs, such as contact details like name, email address, phone numbers, or details like credit card information, etc.

Forms contain special elements called controls like inputbox, checkboxes, radio-buttons, submit buttons, etc. Users generally complete a form by modifying its controls e.g. entering text, selecting items, etc. and submitting this form to a web server for processing.

```
<form>
    <fieldset>
        <legend>Log In</legend>
        <label>Username: <input type="text"></label>
        <label>Password: <input type="password"></label>
        <input type="submit" value="Submit">
    </fieldset>
</form>
```

Most frequently used form attributes are:

Attribute	Description
`name`	The name of the form.
`action`	URL of the program that processes the information submitted via form.
`method`	The HTTP method that the browser uses to submit the form. Possible values are get and post.
`target`	A name or keyword indicating the target page where the result of the script will be displayed. The reserved keywords are _blank, _self, _parent and _top

Input Element

This is the most commonly used element within HTML forms.

It allows you to specify various types of user input fields, depending on the type attribute. An input element can be of type text field, checkbox, password field, radio button, submit button, reset button, etc. and several new input types introduced in HTML5

Text Fields

Text fields are one line areas that allow the user to input text.

Single-line text input controls are created using an <input> element, whose type attribute has a value of text. Here's an example of a single-line text input used to take username

```
<form>
    <label for="username">Username:</label>
    <input type="text" name="username" id="username">
</form>
```

Note: The <label> tag to define labels for <input> elements. If you want your user to enter several lines you should use a <textarea> instead.

Password Field

Password fields are like text fields. The only difference is; characters in a password field are masked i.e. shown as asterisks or dots. This is to prevent others from reading the password on the screen. This is also a single-line text input controls created using an <input> element whose type attribute has a value of password.

```
<form>
    <label for="user-pwd">Password:</label>
    <input type="password" name="user-password" id="user-pwd">
</form>
```

Radio Buttons

Radio buttons are used to let the user select exactly one option from a pre-defined set of options. It is created using an <input> element whose type attribute has a value of radio.

```
<form>
    <input type="radio" name="sex" id="male">
    <label for="male">Male</label>
    <input type="radio" name="sex" id="female">
    <label for="female">Female</label>
</form>
```

Checkboxes

Checkboxes allows the user to select one or more option from a pre-defined set of options. It is created using an <input> element whose type attribute has a value of checkbox.

```
<form>
    <input type="checkbox" name="sports" id="soccer">
    <label for="soccer">Soccer</label>
    <input type="checkbox" name="sports" id="cricket">
    <label for="cricket">Cricket</label>
    <input type="checkbox" name="sports" id="cricket">
    <label for="baseball">Baseball</label>
```

```
        </form>
```

File Select box

The file fields allow a user to browse for a local file and send it as an attachment to the form data. It normally rendered as a text box with a button that enables the user to browse for a file. However, the user can also type the path and name of the file in the text box.

This is also created using an `<input>` element, whose type attribute value is set to `file`.

```
        <form>
            <label for="file-select">Upload:</label>
            <input type="file" name="upload" id="file-select">
        </form>
```

New Input Types in HTML5

HTML5 introduces several new <input> types like email, date, time, color, range, etc. to improve the user experience and to make the forms more interactive. However, if a browser failed to recognize these new input types, it will treat them like a normal text box.

color	date	datetime	datetime-local
email	month	number	range
search	tel	time	url
week			

Input Type Color

The color input type allows the user to select a color from a drop-down color picker and returns the hex value for that color

```
        <form>
            <label>
                Select Color: <input type="color" name="mycolor">
            </label>
        </form>
```

Input Type Date

The `date` input type allows the user to select a date from a drop-down calendar.

```
        <form>
            <label>
                Select Date: <input type="date" name="mydate">
            </label>
        </form>
```

Input Type Datetime

The `datetime` input type allows the user to select a date and time along with time zone.

```
<form>
    <label>
        Date & Time: <input type="datetime" name="mydatetime">
    </label>
</form>
```

Input Type Datetime-local

The `datetime-local` input type allows the user to select a local date and time. The local date and time doesn't provide timezone information

```
<form>
    <label>
        Local Date & Time: <input type="datetime-local" name="mylocaldatetime">
    </label>
</form>
```

Input Type Email

The `email` input type allows the user to enter e-mail address. It is very similar to a standard text input type, but if it used in combination with the required attribute, the browser may look for patterns to ensure a valid e-mail address should be entered.

```
<form>
    <label>
        Email Address: <input type="email" name="myemail" required>
    </label>
</form>
```

The validation for the input `type="email"` is supported by all major browsers like Mozilla Firefox, Google Chrome, Apple Safari, Internet Explorer 10+ and Opera.

Input Type Month

The `month` input type allows the user to select a month and year from a drop-down calendar

```
<form>
    <label>
        Select Month: <input type="month" name="mymonth">
    </label>
</form>
```

Input Type Number

The `number` input type can be used for entering a numerical value. You can also restrict the user to enter only acceptable values using the additional attributes `min`, `max`, and `step`

```
<form>
    <label>
        Select Number: <input type="number" value="1" min="1" max="10" step="0.5"
name="mynumber">
    </label>
</form>
```

The input `type="number"` is supported by all the major browsers like Mozilla Firefox, Google Chrome, Apple Safari, Internet Explorer 10+ and Opera. Internet Explorer however recognized the number but do not provide increment and decrement spin buttons.

Input Type Range

The `range` input type can be used for entering a numerical value within a specified range. It works very similar to `number` input, but it offer a simpler control for entering a number

```
<form>
    <label>
        Select Number: <input type="range" value="1" min="1" max="10" step="0.5"
name="mynumber">
    </label>
</form>
```

The input `type="range"` is supported by all the major browsers like Mozilla Firefox, Google Chrome, Apple Safari, Internet Explorer 10+ and Opera.

Input Type Search

The `search` input type can be used for creating search fields.

A search field typically behaves like a regular text field, but in some browser like Google Chrome and Apple Safari as soon as you start typing in a search box a small cross appears on the right side of the field that lets you quickly clear the search field

```
<form>
    <label>
        Search Website: <input type="search" name="mysearch">
    </label>
</form>
```

Input Type Tel

The `tel` input type can be used for entering a telephone number

```
<form>
    <label>
        Telephone Number: <input type="tel" name="mytelephone" required>
    </label>
</form>
```

Input Type Time

The `time` input type can be used for entering a time.

```
<form>
    <label>
        Select Time: <input type="time" name="mytime">
    </label>
</form>
```

Input Type URL

The url input type can be used for entering web addresses i.e. URL's. You can use the multiple attribute to enter more than one URL. Like type="email", a browser may carry out simple validation on URL fields and present an error message on form submissio

```
<form>
    <label>
        Website URL: <input type="url" name="mywebsite" required>
    </label>
</form>
```

Input Type Week

The week input type allows the user to select a week and year from a drop-down calendar

```
<form>
    <label>
        Select Week: <input type="week" name="myweek">
    </label>
</form>
```

Textarea

Textarea is a multiple-line text input control that allows a user to enter more than one line of text. Multi-line text input controls are created using an `<textarea>` element.

```
<form>
    <label for="address">Address:</label>
    <textarea rows="3" cols="30" name="address" id="address"></textarea>
</form>
```

Select Boxes

A select box is a dropdown list of options that allows user to select one or more option from a pull-down list of options. Select box is created using the `<select>` element and `<option>` element. The option elements within the `<select>` element define each list item.

```
<form>
    <label for="city">City:</label>
    <select name="city" id="city">
        <option value="sydney">Sydney</option>
        <option value="melbourne">Melbourne</option>
        <option value="cromwell">Cromwell</option>
    </select>
</form>
```

Submit and Reset Buttons

A submit button is used to send the form data to a web server. When submit button is clicked the form data is sent to the file specified in the form's action attribute to

process the submitted data. A reset button resets all the forms control to default values.

```
<form action="action.php" method="post" id="users">
      <label for="first-name">First Name:</label>
      <input type="text" name="first-name" id="first-name">
      <input type="submit" value="Submit">
      <input type="reset" value="Reset">
</form>
```

Note: you can also create buttons using the <button> element. Buttons created with the <button> element function just like buttons created with the input element, but they offer richer rendering possibilities.

HTML5 form attributes

There are 14 new attributes that we'll be looking at in this article.

placeholder	autofocus	autocomplete	required
pattern	list	multiple	novalidate
formnovalidate	form	formaction	formenctype
formmethod	formtarget		

The readonly Attribute

The readonly attribute specifies that the input field is read only (cannot be changed):

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="John" readonly>
</form>
```

The disabled Attribute

The disabled attribute specifies that the input field is disabled.

A disabled input field is unusable and un-clickable, and its value will not be sent when submitting the form:

```
<form action="">
First name:<br>
```

```
<input type="text" name="firstname" value="John" disabled>
</form>
```

The size Attribute

The size attribute specifies the size (in characters) for the input field:

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="John" size="40">
</form>
```

The maxlength Attribute

The maxlength attribute specifies the maximum allowed length for the input field

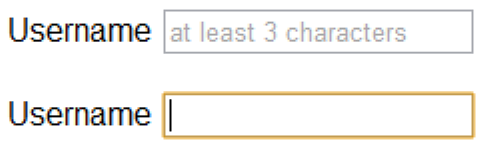
```
<form action="">
First name:<br>
<input type="text" name="firstname" maxlength="10">
</form>
```

With a maxlength attribute, the input field will not accept more than the allowed number of characters.

The maxlength attribute does not provide any feedback. If you want to alert the user, you must write JavaScript code

placeholder

placeholder attribute allows us to set placeholder text as we would currently do in HTML4 with the value attribute. It should only be used for short descriptions. For anything longer, use the title attribute. The difference from HTML4 is that the text is only displayed when the field is empty and hasn't received focus. Once the field receives focus (e.g., you click or tab to the field), and you begin to type, the text simply disappears



```
<input type="text" name="user-name" id="user-name" placeholder="at least 3 characters">
```

autofocus

Adding it to an input automatically focuses that field when the page is rendered. As with placeholder, autofocus is something that we used JavaScript for in the past.

```
<input type="text" name="first-name" id="first-name" autofocus>
```

Note: Several new HTML5 form attributes are Boolean attributes. This just means they're set if they're present and not set if they're absent. They can be written several ways in HTML5.

```
autofocus
autofocus=""
autofocus="autofocus"
```

autocomplete

The **autocomplete** attribute specifies whether a form or input field should have autocomplete on or off.

When autocomplete is on, the browser automatically complete the input values based on values that the user has entered before.

Tip: It is possible to have autocomplete "on" for the form, and "off" for specific input fields, or vice versa.

The autocomplete attribute works with <form> and the following <input> types: text, search, url, tel, email, password, datepickers, range, and color.

The autocomplete attribute helps users to complete forms based on earlier input. The attribute has been around since IE5.5 but has finally been standardized as part of HTML5. The default state is set to on. This means that generally we won't have to use it. However, if you want to insist that a form field be entered each time a form is completed (as opposed to the browser auto filling the field), you would implement it like so:

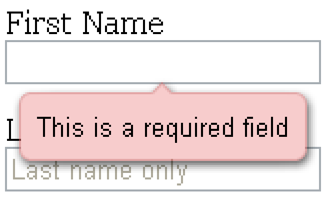
```
<form action="/action_page.php" autocomplete="on">
  First name:<input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  E-mail: <input type="email" name="email" autocomplete="off"><br>
  <input type="submit">
</form>
```

The autocomplete state on a field overrides any autocomplete state set on the containing form element.

required

By adding it to a form field, the browser requires the user to enter data into that field before submitting the form. This replaces the basic form validation currently implemented with JavaScript, making things a little more usable and saving us a little more development time. required is a Boolean attribute, like autofocus. Let's see it in action.

```
<input type="text" id="given-name" name="given-name" required>
```



pattern

The pattern attribute is likely to get a lot of developers very excited. It specifies a JavaScript regular expression for the field's value to be checked against. pattern makes it easy for us to implement specific validation for product codes, invoice numbers, and so on. The possibilities for pattern are wide-ranging, and this is just one simple example using a product number.

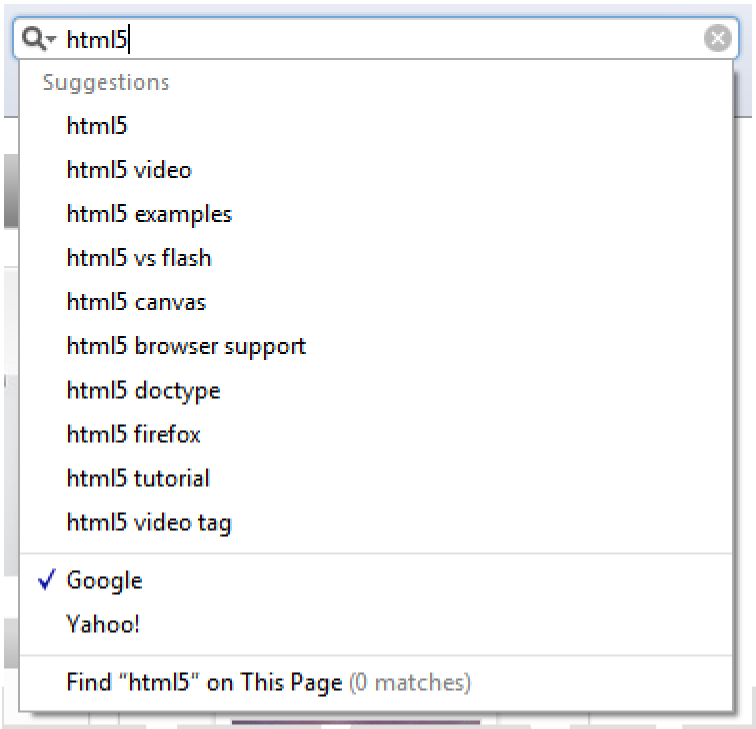
```
<label>Product Number:
<input pattern="[0-9][A-Z]{3}" name="product" type="text" title="Single digit
followed by three uppercase letters."/>
</label
```

This pattern prescribes that the product number should be a single digit [0-9] followed by three uppercase letters [A-Z]{3}. For more examples, the HTML5 Pattern website lists common regex style patterns to help get you started.

list and the datalist element

The list attribute enables the user to associate a list of options with a field. The value of the list attribute must be the same as the ID of a datalist element that resides in the same document.

The datalist element is new in HTML5 and represents a predefined list of options for form controls. It works in a similar way to the in-browser search boxes that autocomplete as you type (see Figure 4).



```
<label>Your favorite fruit:
<datalist id="fruits">
  <option value="Blackberry">Blackberry</option>
  <option value="Blackcurrant">Blackcurrant</option>
  <option value="Blueberry">Blueberry</option>
  <!-- … -->
</datalist>
If other, please specify:
  <input type="text" name="fruit" list="fruits">
</label>
```

By adding a select element inside the datalist you can provide superior graceful degradation than by simply using an option element. This is an elegant markup pattern designed by Jeremy Keith that adheres perfectly with HTML5's principle of degrading gracefully.

```
<label>Your favorite fruit:
<datalist id="fruits">
  <select name="fruits">
    <option value="Blackberry">Blackberry</option>
    <option value="Blackcurrant">Blackcurrant</option>
    <option value="Blueberry">Blueberry</option>
    <!-- … -->
  </select>
If other, please specify:
</datalist>
  <input type="text" name="fruit" list="fruits">
</label>
```

multiple

We can take our lists and datalists one step further by applying the Boolean attribute **multiple** to allow more than one value to be entered from the datalist. Here is an example.

```
  <label>Your favorite fruit:
  <datalist id="fruits">
    <select name="fruits">
      <option value="Blackberry">Blackberry</option>
      <option value="Blackcurrant">Blackcurrant</option>
      <option value="Blueberry">Blueberry</option>
      <!-- … -->
    </select>
  If other, please specify:
  </datalist>
    <input type="text" name="fruit" list="fruits" multiple>
  </label>
```

multiple isn't exclusively for use with datalists, though. A further example for multiple might be for email addresses when sending items to friend or the attachment of files, as shown here:

```
  <label>Upload files:
  <input type="file" multiple name="upload"></label>
```

novalidate and formnovalidate

The novalidate and formnovalidate attributes indicate that the form shouldn't be validated when submitted. They are both Boolean attributes. formnovalidate can be applied to submit or image input types. The novalidate attribute can be set only on the form element.

An example use case for the formnovalidate attribute could be on a "save draft" button, where the form has fields that are required for submitting the draft but aren't required for saving the draft. novalidate would be used in cases where you don't want to validate the form but do want to take advantage of the more useful user interface enhancements that the new input types offer.

The following example shows how to use formnovalidate:

```
<form action="process.php">
  <label for="email">Email:</label>
  <input type="text" name="email"value="gordo@example.com">
  <input type="submit" formnovalidate value="Submit">
</form>
```

And this example shows how to use novalidate:

```
<form action="process.php" novalidate>
  <label for="email">Email:</label>
  <input type="text" name="email"value="gordo@example.com">
  <input type="submit" value="Submit">
</form>
```

The formaction Attribute

The formaction attribute specifies the URL of a file that will process the input control when the form is submitted. The formaction attribute overrides the action attribute of the <form> element.

The formaction attribute is used with type="submit" and type="image".

```
<form action="/action_page.php">
  First name: <input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  <input type="submit" value="Submit"><br>
  <input type="submit" formaction="/action_page2.php"
  value="Submit as admin">
</form>
```

The formenctype Attribute

The **formenctype** attribute specifies how the form data should be encoded when submitted (only for forms with method="post").

The formenctype attribute overrides the enctype attribute of the <form> element.

The formenctype attribute is used with type="submit" and type="image".

Send form-data that is default encoded (the first submit button), and encoded as "multipart/form-data" (the second submit button):

```
<form action="/action_page_binary.asp" method="post">
  First name: <input type="text" name="fname"><br>
  <input type="submit" value="Submit">
  <input type="submit" formenctype="multipart/form-data"
  value="Submit as Multipart/form-data">
</form>
```

The formmethod Attribute

The **formmethod** attribute defines the HTTP method for sending form-data to the action URL.

The formmethod attribute overrides the method attribute of the <form> element.

The formmethod attribute can be used with type="submit" and type="image".

The second submit button overrides the HTTP method of the form:

```
<form action="/action_page.php" method="get">
  First name: <input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  <input type="submit" value="Submit">
  <input type="submit" formmethod="post" formaction="action_page_post.asp"
  value="Submit using POST">
</form>
```

Video

Embedding Video in HTML Document

Inserting video onto a web page is not relatively easy, because browsers did not have a uniform standard for defining embedded media files.

Using the HTML5 video Element

The newly introduced HTML5 `<video>` element provides a standard way to embed video in web pages. However, the video element is relatively new, but it works in most of the modern web browsers. The following example simply inserts a video into the HTML5 document, using the browser default set of controls, with one source.

Video Formats Supported by different browsers

Browser	MP4	WebM	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	NO	NO
Opera	NO	YES	YES

Format	MIME-type
MP4	video/mp4
WebM	video/webm
Ogg	video/ogg

```
<video controls="controls" src="shuttle.mp4">
    Your browser does not support the HTML5 Video element.
</video>
```

A video, using the browser default set of controls, with alternative sources.

```
<video controls="controls">
    <source src="shuttle.mp4" type="video/mp4">
    <source src="shuttle.ogv" type="video/ogg">
    Your browser does not support the HTML5 Video element.
</video>
```

Properties:

currentTime	Gets or sets the current playback position in seconds
volume	Gets or sets the current volume level for the video
muted	Gets or sets the mute state
playbackRate	Gets or sets the playback rate, where 1 is normal speed forward
currentSrc	Returns the current video source file the browser is playing
videoWidth & videoHeight	Returns the actual dimensions of the video, not the video element size

Methods:

load()	Loads the video and reset the play head to the beginning of the video
play()	Plays the video from it's current location
pause()	Pauses the video at the current location
canPlayType(format)	Tests to see whether the browser can play a specific type of video, for example 'video/webm;codecs="vp8, vorbis"' The browser will return: **probably** - if it's most likely the video file can be played **maybe** - if the video might be playable **[empty string]** - if the video file is not playable

Events:

canplaythrough	Fired when enough data is available that the browser believes it can play the video completely without interruption
ended	Fired when the video has finished playing
error	Fired if an error occurs
playing	Fired when the video starts playing, for the first time, after being paused or when restarting
progress	Fired periodically to indicate the progress of downloading the video
waiting	Fired when an action is delayed pending the completion of another action
loadedmetadata	Fired when the browser has finished loading the metadata for the video and all attributes have been populated

```
    <html lang="en" xmlns="http://www.w3.org/1999/xhtml"> <head>
        <meta charset="utf-8" />
        <title>Video Demo</title>
    </head>
    <body>
        <div style="text-align:center">
            <button onclick="playPause()">Play/Pause</button>
            <button onclick="reload()">Reload</button>
            <button onclick="makeBig()">Big</button>
            <button onclick="makeSmall()">Small</button>
            <button onclick="makeNormal()">Normal</button>
            <br>
            <video id="video1" width="420" controls>
                 <source src="Intro.mp4" type="video/mp4">
                 <source src="Intro.ogg" type="video/ogg">
                 Your browser does not support HTML5 video
           </video>
        </div>
  <script>
      var myVideo = document.getElementById("video1");

      function playPause() {
    if (myVideo.paused)
          myVideo.play();
    else
          myVideo.pause();
      }

      function reload() {
          alert("Can play MP4: " + myVideo.canPlayType("video/mp4"));
           myVideo.load();
      }

      function makeBig() {
          myVideo.width = 560;
      }

      function makeSmall() {
          myVideo.width = 320;
      }

      function makeNormal() {
          myVideo.width = 420;
      }
    </script>
</body>
</html>
```

Using the object Element

The `<object>` element is used to embed different kinds of media files into an HTML document. Initially, this element was used to insert ActiveX controls, but according to the specification, an object can be any media object such as video, audio, Java applets, ActiveX, document (HTML, PDF, Word, etc.), Flash animations or even images.

The following code fragment embeds a Flash video into a web page.

```
<object data="blur.swf" width="400px" height="200px"></object>
```

Only browsers or applications that support Flash will play this video.

Warning: The <object> element is not supported widely and very much depends on the type of the object that's being embedded. Other methods could be a better choice in many cases. iPad and iPhone cannot display Flash videos.

Using the embed Element

The <embed> element is used to embed multimedia content into an HTML document.

The following code fragment embeds a Flash video into a web page.

```
<embed src="blur.swf" width="400px" height="200px">
```

Warning: However the <embed> element is very well supported in current browsers and defined as standard in HTML5, but your video might not played due to lack of browser support for Flash or unavailability of plugins.

Embedding the YouTube Videos

This is the easiest and popular way to embed videos files in the web pages. Just upload the video on YouTube and insert HTML code to display that video in your web page.

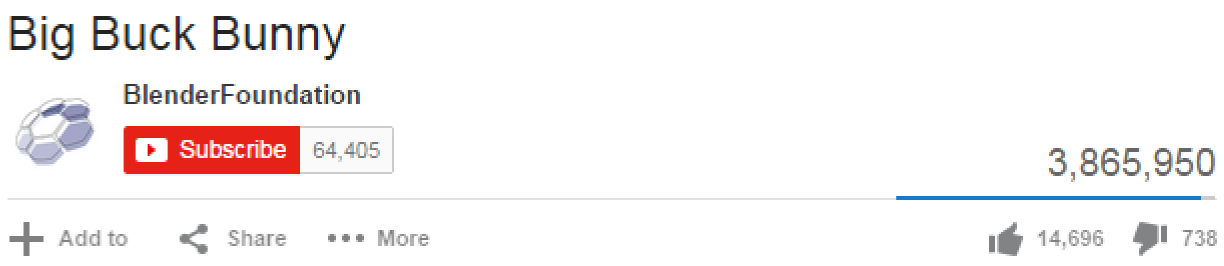
Here's a live example followed by the explanation of whole process:

Step 1: Upload video

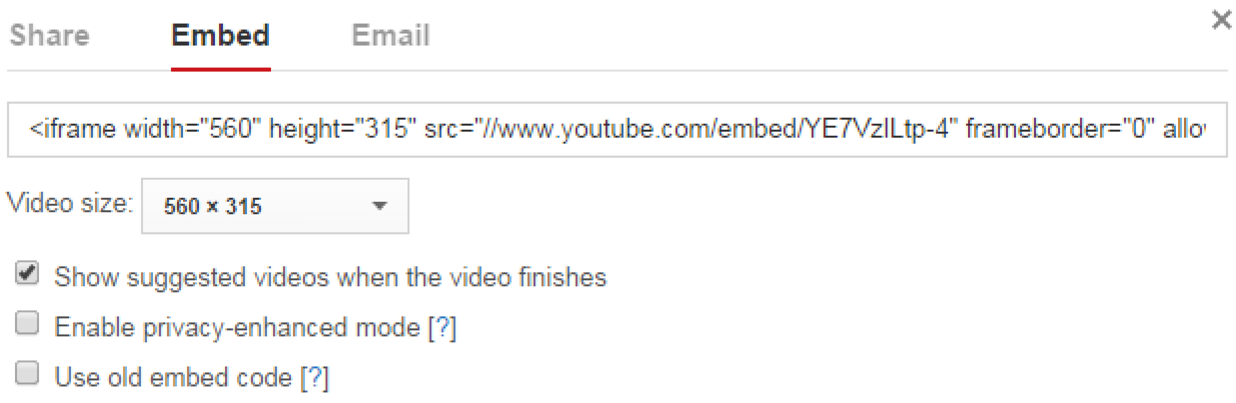
Go to YouTube Upload video page and follow the instructions to upload your video.

Step 2: Creating the HTML Code to embed the video

When you open your uploaded video in YouTube you will see something like this on the bottom of video. Browse and open your uploaded video in YouTube. Now look for the share button, located just below the video as shown in the figure.



When you click the Share button, a share panel will open displaying some more buttons. Now click on the Embed button, it will generate the HTML code to directly embed the video into the web pages. Just copy and paste that code into your HTML document where you want to display the video and you're all set. By default video embedded inside an iframe.



```
<iframe width="560" height="315" src="//www.youtube.com/embed/YE7VzlLtp-4" frameborder="0" allo
```

Video size: 560 × 315

- [✓] Show suggested videos when the video finishes
- [ ] Enable privacy-enhanced mode [?]
- [ ] Use old embed code [?]

You can further customize this embed code such as changing the video size by selecting the customization option given just below the embed-code input box. Here's an example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>YouTube Video</title>
</head>
<body>
    <iframe width="560" height="315" src="//www.youtube.com/embed/YE7VzlLtp-4"
frameborder="0" allowfullscreen></iframe>
</body>
</html>
```