**WWN Development Document**

HTML Tutorials, Useful in Creating WWN

Word’s Navigation pane shows the table-of-contents (View : Show : Navigation pane).

* Contents:
  + HTML references and tutorials

This document was created by the WWN author for his own use in developing WWN. It is included in the WWN repo, as other developers may find it useful.

--------------------

For the sections with tutorials, some of the notes are copied directly from the tutorials. The copyright for the Mozilla tutorials is:

*Portions of this content are ©1998–2023 by individual mozilla.org contributors. Content available under a Creative Commons license.*

# References

* HTML elements reference - HTML: HyperText Markup Language | MDN
  + <https://developer.mozilla.org/en-US/docs/Web/HTML/Element>

## Tools

* All HTML linters - Mega-Linter
  + <https://nvuillam.github.io/mega-linter/descriptors/html/>

## Testing and debugging

* What are browser developer tools? - Learn web development | MDN
  + <https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_are_browser_developer_tools>
  + The Inspector: DOM explorer and CSS editor
  + The JavaScript debugger
  + The JavaScript console
* Running your HTML page through the Markup Validation Service
  + The W3C Markup Validation Service
    - <https://validator.w3.org/>

# HTML tutorial

* Introduction to HTML - Learn web development | MDN
  + <https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML>

## Getting started with HTML

* Getting started with HTML - Learn web development | MDN
  + <https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Getting_started>
* Block versus inline elements
  + **Block-level elements** form a visible block on a page. A block-level element appears on a new line following the content that precedes it. Any content that follows a block-level element also appears on a new line. Block-level elements are usually structural elements on the page. For example, a block-level element might represent headings, paragraphs, lists, navigation menus, or footers. A block-level element wouldn't be nested inside an inline element, but it might be nested inside another block-level element.
  + **Inline elements** are contained within block-level elements, and surround only small parts of the document’s content (not entire paragraphs or groupings of content). An inline element will not cause a new line to appear in the document. It is typically used with text, for example an <a> element creates a hyperlink, and elements such as <em> or <strong> create emphasis.
  + The terms block and inline, as used in this article, should not be confused with the types of CSS boxes that have the same names.
* Empty elements
  + Not all elements follow the pattern of an opening tag, content, and a closing tag.
  + Some elements consist of a single tag, which is typically used to insert/embed something in the document. For example, the <img> element embeds an image file onto a page
* Elements can also have attributes.



<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>My test page</title>

</head>

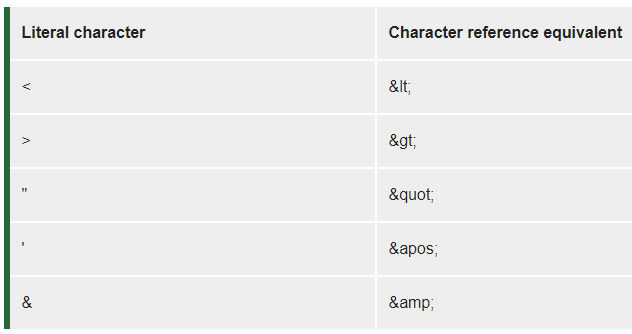
<body>

<p>This is my page</p>

</body>

</html>

* doctype is a historical artifact
* <html></html>: The **<html> element**.
  + This element wraps all the content on the page.
  + It is sometimes known as the **root element**.
* <head></head>: The **<head> element**.
  + This element acts as a container for everything you want to include on the HTML page, that isn't the content the page will show to viewers.
  + This includes keywords and a page description that would appear in search results, CSS to style content, character set declarations, and more.
* <meta charset="utf-8">: This element specifies the character set for your document to UTF-8, which includes most characters from the vast majority of human written languages. With this setting, the page can now handle any textual content it might contain. There is no reason not to set this, and it can help avoid some problems later.
* <title></title>: The <title> element. This sets the title of the page, which is the title that appears in the browser tab the page is loaded in. The page title is also used to describe the page when it is bookmarked.
* <body></body>: The <body> element. This contains all the content that displays on the page, including text, images, videos, games, playable audio tracks, or whatever else.
* In HTML, the characters <, >,",' and & are special characters. They are parts of the HTML syntax itself. So how do you include one of these special characters in your text?



## What’s in the head? Metadata in HTML

* What’s in the head? Metadata in HTML
  + <https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/The_head_metadata_in_HTML>
* Many <meta> elements include name and content attributes:
  + **name** specifies the type of meta element it is; what type of information it contains.
  + **content** specifies the actual meta content.

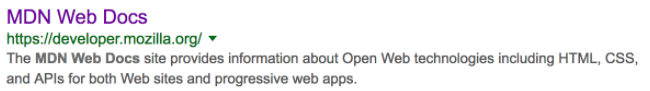
<meta name="author" content="Chris Mills">

<meta name="description" content="The MDN Web Docs Learning Area aims to provide

complete beginners to the Web with all they need to know to get

started with developing web sites and applications.">

* Specifying a **description** that includes keywords relating to the content of your page is useful as it has the potential to make your page appear higher in relevant searches performed in search engines
* Now search for "MDN Web Docs" in your favorite search engine (We used Google.) You'll notice the description <meta> and <title> element content used in the search result



* Many <meta> features just aren't used any more. For example, the keyword <meta> element (<meta name="keywords" content="fill, in, your, keywords, here">)
* other types of metadata, too. A lot of the features you'll see on websites are proprietary creations, designed to provide certain sites (such as social networking sites) with specific pieces of information they can use.
  + For example, Open Graph Data is a metadata protocol that Facebook invented to provide richer metadata for websites.
    - when you link to MDN Web Docs on facebook, the link appears along with an image and description: a richer experience for users
  + Twitter also has its own similar proprietary metadata called Twitter Cards
* Adding custom icons to your site
  + you can add references to custom icons in your metadata, and these will be displayed in certain contexts. The most commonly used of these is the **favicon** (short for "favorites icon", referring to its use in the "favorites" or "bookmarks" lists in browsers).
  + There are lots of other icon types to consider these days as well...
* The <link> element should always go inside the head of your document.
  + <link rel="stylesheet" href="my-css-file.css">
* The <script> element should also go into the head, and should include a src attribute containing the path to the JavaScript you want to load, and **defer**, which basically instructs the browser to load the JavaScript at the same time as the page's HTML.
  + This is useful as it makes sure that the HTML is all loaded before the JavaScript runs, so that you don't get errors resulting from JavaScript trying to access an HTML element that doesn't exist on the page yet. There are actually a number of ways to handle loading JavaScript on your page, but this is the most foolproof one to use for modern browsers (for others, read Script loading strategies).
  + <script src="my-js-file.js" defer></script>
* Setting the primary language of the document
  + <html lang="en-US">

## Creating hyperlinks

* Adding supporting information with the title attribute
  + This gives us the following result and hovering over the link displays the title as a tooltip.

<a href="https://www.mozilla.org/en-US/"

title="The best place to find more information about Mozilla's

mission and how to contribute">the Mozilla homepage</a>

* Block level links
  + almost any content can be made into a link, even block-level elements.
  + How to make an image a clickable link:
    - If you have an image you want to make into a link, use the <a> element and reference the image file with the <img> element.

<a href="https://www.mozilla.org/en-US/">

<img src=" https://www.mozilla.org/media/protocol/img/logos/mozilla/black.40d1af88c248.svg" alt="mozilla logo that links to the mozilla homepage">

</a>

* Absolute versus relative URLs
  + relative URL: Points to a location that is relative to the file you are linking from,
* Link best practices
  + Use clear link wording
    - Search engines use link text to index target files, so it is a good idea to include keywords in your link text to effectively describe what is being linked to.
* Other tips:
  + Don't repeat the URL as part of the link text — URLs look ugly, and sound even uglier when a screen reader reads them out letter by letter.
  + Don't say "link" or "links to" in the link text — it's just noise. Screen readers tell people there's a link. Visual users will also know there's a link, because links are generally styled in a different color and underlined (this convention generally shouldn't be broken, as users are used to it).
  + Keep your link text as short as possible — this is helpful because screen readers need to interpret the entire link text.
  + Minimize instances where multiple copies of the same text are linked to different places. This can cause problems for screen reader users, if there's a list of links out of context that are labeled "click here", "click here", "click here".
* Use the **download attribute** when linking to a download
  + When you are linking to a resource that's to be downloaded rather than opened in the browser, you can use the download attribute to provide a default save filename. Here's an example with a download link to the latest Windows version of Firefox:

<a href="https://download.mozilla.org/?product=firefox-latest-ssl&os=win64&lang=en-US"

download="firefox-latest-64bit-installer.exe">

Download Latest Firefox for Windows (64-bit) (English, US)

</a>

* Active learning: creating a navigation menu
* E-mail links
  + Specifying details
    - In addition to the email address, you can provide other information. In fact, any standard mail header fields can be added to the mailto URL you provide. The most commonly used of these are "subject", "cc", and "body" (which is not a true header field, but allows you to specify a short content message for the new email). Each field and its value is specified as a query term.

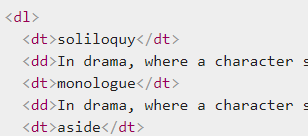
<a href="mailto:nowhere@mozilla.org?cc=name2@rapidtables.com&bcc=name3@rapidtables.com&subject=The%20subject%20of%20the%20email&body=The%20body%20of%20the%20email">

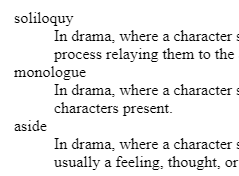
Send mail with cc, bcc, subject and body

</a>

## Advanced text formatting

* Description lists





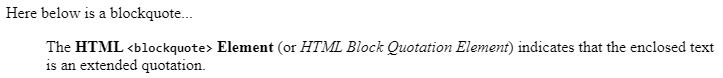
* Blockquotes
  + If a section of block level content (be it a paragraph, multiple paragraphs, a list, etc.) is quoted from somewhere else, you should wrap it inside a <blockquote> element to signify this, and include a URL pointing to the source of the quote inside a cite attribute.

<blockquote cite="https://developer.mozilla.org/en-US/docs/Web/HTML/Element/blockquote">

<p>The <strong>HTML <code>&lt;blockquote&gt;</code> Element</strong> (or <em>HTML Block

Quotation Element</em>) indicates that the enclosed text is an extended quotation.</p>

</blockquote>



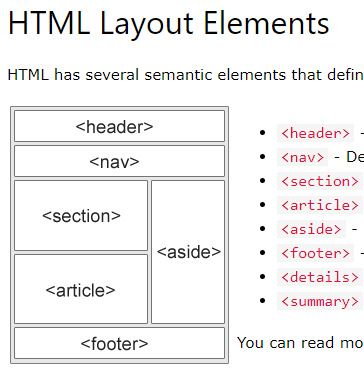
* Abbreviations
  + Another fairly common element you'll meet when looking around the Web is <abbr> — this is used to wrap around an abbreviation or acronym, and provide a full expansion of the term (included inside a title attribute.) Let's look at a couple of examples:

<p>We use <abbr title="Hypertext Markup Language">HTML</abbr> to structure our web documents.</p>



## Document and website structure

* Document and website structure - Learn web development | MDN
  + <https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Document_and_website_structure>
* HTML for structuring content
  + HTML provides dedicated tags that you can use to represent such sections, for example:
    - header: <header>.
    - navigation bar: <nav>.
    - main content: <main>,
      * with various content subsections represented by <article>, <section>, and <div> elements.
    - sidebar: <aside>; often placed inside <main>.
    - footer: <footer>
* HTML Layout Elements and Techniques
  + <https://www.w3schools.com/html/html_layout.asp>



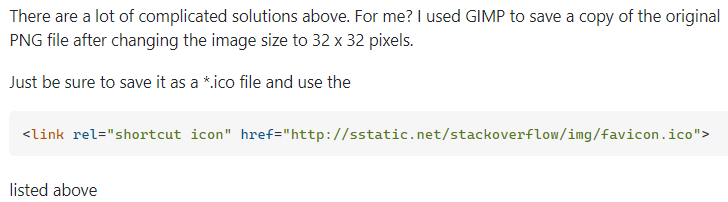
* HTML layout elements in more detail
* <main> is for content unique to this page.
  + Use <main> only once per page, and put it directly inside <body>. Ideally this shouldn't be nested within other elements.
* <article> encloses a block of related content that makes sense on its own without the rest of the page (e.g., a single blog post).
* <section> is similar to <article>, but it is more for grouping together a single part of the page that constitutes one single piece of functionality (e.g., a mini map, or a set of article headlines and summaries), or a theme.
  + It's considered best practice to begin each section with a heading; also note that you can break <article>s up into different <section>s, or <section>s up into different <article>s, depending on the context.
* <aside> contains content that is not directly related to the main content but can provide additional information indirectly related to it (glossary entries, author biography, related links, etc.).
* <header> represents a group of introductory content.
  + If it is a child of <body> it defines the global header of a webpage, but if it's a child of an <article> or <section> it defines a specific header for that section (try not to confuse this with titles and headings).
* <nav> contains the main navigation functionality for the page. Secondary links, etc., would not go in the navigation.
* <footer> represents a group of end content for a page.
* Example HTML is provided
  + My Fiddle: <https://jsfiddle.net/jimyuill/L587rbp4/2/>
  + NOTE: it's not clear if these section-type tags provide any formatting, perhaps not
    - QUESTION: so, why use them?
* Non-semantic wrappers
  + Sometimes you'll come across a situation where you can't find an ideal semantic element to group some items together or wrap some content.
    - Sometimes you might want to just group a set of elements together to affect them all as a single entity with some CSS or JavaScript.
    - For cases like these, HTML provides the <div> and <span> elements.
    - You should use these preferably with a suitable class attribute, to provide some kind of label for them so they can be easily targeted.
  + <div> is a block level non-semantic element,
    - which you should only use if you can't think of a better semantic block element to use, or don't want to add any specific meaning.
* Line breaks and horizontal rules
  + <hr> elements create a horizontal rule in the document that denotes a thematic change in the text (such as a change in topic or scene). Visually it just looks like a horizontal line.

## Debugging HTML

* running your HTML page through the Markup Validation Service
  + The W3C Markup Validation Service
    - <https://validator.w3.org/>

## Favicon (web-page icon)

* Tutorials
  + How to Add Image in the Title Bar
    - <https://www.w3docs.com/snippets/html/how-to-add-an-image-in-the-title-bar.html>
  + Add Favicon: A Beginner's Guide to Add an Icon to Your Website
    - <https://www.hostinger.com/tutorials/how-to-add-favicon-to-website>
* The ultimate favicon generator - Favic-o-Matic
  + <https://favicomatic.com/>
* Stackoverflow
  + html - How can I add a tab icon? - Stack Overflow
    - <https://stackoverflow.com/questions/41906095/how-can-i-add-a-tab-icon>
  + html - How to add a browser tab icon (favicon) for a website? - Stack Overflow
    - <https://stackoverflow.com/questions/4888377/how-to-add-a-browser-tab-icon-favicon-for-a-website>
    - Notes:
      * I used 32x32, per this post, but not this <link>



* Use .ico format
  + favicon.png vs favicon.ico - why should I use PNG instead of ICO? - Stack Overflow
    - <https://stackoverflow.com/questions/1344122/favicon-png-vs-favicon-ico-why-should-i-use-png-instead-of-ico>

### Using gimp to create icon (.ico format)

* create 32x32 canvas and make color green
* use Text tool, and it creates another layer
  + References
    - Adding Text with GIMP
      * <https://www.quackit.com/web_graphics/gimp/tutorial/adding_text_with_gimp.cfm>
  + choose font and size
    - Use Font-selection window/dialog
      * 3.7. Fonts Dialog
        + <https://docs.gimp.org/en/gimp-font-dialog.html>
  + and size textbox to canvas using small squares
  + choose center text alignment
    - Discover How To Center Text In Gimp
      * <https://parkerphotographic.com/how-to-center-text-in-gimp/>
* Save as gimp format file
* Make both layers visible, and merge them (Image : Merge)
* Save As different gimp format file
* Export as ".ico" file

### Installing on web-server

* Put favicon.ico on web-server root
  + Browsers will use it by default, without <link> in HTML <head>
  + Chrome apparently does not use it if the web-page is local, or if the web-page has been accessed before (need to restart Chrome)
    - HTML favicon won't show on google chrome - Stack Overflow
      * <https://stackoverflow.com/questions/13780402/html-favicon-wont-show-on-google-chrome>
    - html - Favicon not showing up in Google Chrome - Stack Overflow
      * <https://stackoverflow.com/questions/16375592/favicon-not-showing-up-in-google-chrome>
* This HTML is probably good for <head>

<link rel="icon" href="/favicon.ico" type="image/x-icon">