RÉFÉRENCE HTML

<https://developer.mozilla.org/en-US/docs/Learn/HTML>

Liste des Block-level elements:

<https://developer.mozilla.org/en-US/docs/Web/HTML/Block-level_elements>

Liste des inline elements:

<https://developer.mozilla.org/en-US/docs/Web/HTML/Inline_elements>

Pour valider du code HTML:

<https://validator.w3.org/>

Publier web page with Github:

https://developer.mozilla.org/en-US/docs/Learn/Common\_questions/Using\_Github\_pages

**Note**: If you are working on a computer/tablet/other device where you don't have the ability to create your own files, you could try out (most of) the code examples in an online coding program such as [JSBin](http://jsbin.com/) or [Thimble](https://thimble.mozilla.org/).

Open your command prompt (Windows)/terminal (OS X/Linux). To check Python is installed, enter the following command:

python –V

  This should return a version number. If this is OK, navigate to the directory that your example is inside, using the cd command.

# include the directory name to enter it, for example

cd Desktop

# use two dots to jump up one directory level if you need to

cd ..

  Enter the command to start up the server in that directory:

# On Mac and Linux

python -m SimpleHTTPServer

# On Windows

python -m http.server

  By default, this will run the contents of the directory on a local web server, on port 8000. You can go to this server by going to the URL localhost:8000 in your web browser. Here you'll see the contents of the directory listed — click the HTML file you want to run.

**Note**: If you already have something running on port 8000, you can choose another port by running the server command followed by an alternative port number, e.g. python -m SimpleHTTPServer 7800. You can then access your content at localhost:7800.

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

**Running server-side languages locally**

Python's SimpleHTTPServer module is useful, but it doesn't know how to run code written in languages such as PHP or Python. To handle that you'll need something more — exactly what you'll need depends on the server-side language you are trying to run. Here are a few examples:

* To run Python server-side code, you'll need to use a Python web framework. You can find out how to use the Django framework by reading [Django Web Framework (Python)](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django). [Flask](http://flask.pocoo.org/) is also a good (slightly less heavyweight) alternative to Django. To run this you'll need to [install Python/PIP](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/development_environment#Installing_Python_3), then install Flask using pip3 install flask. At this point you should be able to run the Python Flask examples using for example python3 python-example.py, then navigating to localhost:5000 in your browser.
* To run Node.js (JavaScript) server-side code, you'll need to use raw node or a framework built on top of it. Express is a good choice — see [Express Web Framework (Node.js/JavaScript)](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs).
* To run PHP server-side code, you'll need a server setup that can interpret PHP. Good options for local PHP testing are [MAMP](https://www.mamp.info/en/downloads/) (Mac and Windows) , [AMPPS](http://ampps.com/download) (Mac, Windows, Linux) and [LAMP](https://www.linux.com/learn/easy-lamp-server-installation) (Linux, Apache, MySQL, and PHP/Python/Perl). These are complete packages that create local setups to allow you to run the Apache server, PHP, and MySQL databases.

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

To choose a color, go to [the Color Picker](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Colors/Color_picker_tool)

#33DAF4

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

image: <https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwjuhbXArvPVAhWL8YMKHSUlBDQQjRwIBw&url=http%3A%2F%2Ftampabaybeersyndicate.com%2F&psig=AFQjCNGlPuB8eS2pMp_ZfYhT6hWKEGDBzw&ust=1503784071426593>

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

Font: Google fonts

**Embed Font**

To embed your selected fonts into a webpage, copy this code into the <head> of your HTML document.

<link href="https://fonts.googleapis.com/css?family=Droid+Sans" rel="stylesheet">

**Specify in CSS**

Use the following CSS rules to specify these families:

font-family: 'Droid Sans', sans-serif;

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

Some general rules for file paths:

* To link to a target file in the same directory as the invoking HTML file, just use the filename, e.g. my-image.jpg.
* To reference a file in a subdirectory, write the directory name in front of the path, plus a forward slash, e.g. subdirectory/my-image.jpg.
* To link to a target file in the directory **above** the invoking HTML file, write two dots. So for example, if index.html was inside a subfolder of test-site and my-image.jpg was inside test-site, you could reference my-image.jpg from index.html using ../my-image.jpg.
* You can combine these as much as you like, for example ../subdirectory/another-subdirectory/my-image.jpg.

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



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, class is the attribute name, and editor-note is the attribute value. The class attribute allows you to give the element an identifier that can be later used to target the element with style information and other things.

An attribute should always have:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name, followed by an equals sign.
3. Opening and closing quote marks wrapped around the attribute value.

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

<!DOCTYPE html> — the doctype. In the mists of time, when HTML was young (about 1991/2), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However, these days no one really cares about them, and they are really just a historical artifact that needs to be included for everything to work right. For now, that's all you need to know.

<html></html> — the [<html>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/html) element. This element wraps all the content on the entire page, and is sometimes known as the root element.

<head></head> — the [<head>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head) element. This element acts as a container for all the stuff you want to include on the HTML page that *isn't* the content you are showing to your page's viewers. This includes things like [keywords](https://developer.mozilla.org/en-US/docs/Glossary/keyword) and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.

<body></body> — the [<body>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/body) element. This contains *all* the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.

<meta charset="utf-8"> — this element sets the character set your document should use to UTF-8, which includes most characters from the vast majority of human written languages. Essentially it can now handle any textual content you might put on it. There is no reason not to set this, and it can help avoid some problems later on.

<title></title> — the [<title>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/title) element. This sets the title of your page, which is the title that appears in the browser tab the page is loaded in. It is also used to describe the page when you bookmark/favorite it.

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

Anything in a CSS document between /\* and \*/ is a **CSS comment**

**-------------**

**Nom de l'Auteur et description du site**

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

<meta name="description" content="The MDN 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.">

**--------------**

BOXES in CSS

Not surprisingly, CSS layout is based principally on the *box model.* Each of the blocks taking up space on your page has properties like this:

* padding, the space just around the content (e.g., around paragraph text)
* border, the solid line that sits just outside the padding
* margin, the space around the outside of the element
* width (of an element)
* background-color, the color behind an element's content and padding
* color, the color of an element's content (usually text)
* text-shadow: sets a drop shadow on the text inside an element
* display: sets the display mode of an element (don't worry about this yet)

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

Entity references: including special characters in HTML

| Literal character | Character reference equivalent |
| --- | --- |
| < | &lt; |
| > | &gt; |
| " | &quot; |
| ' | &apos; |
| & | &amp; |

autres références: <https://dev.w3.org/html5/html-author/charref>

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

To turn a section of content inside your HTML file into a comment, you need to wrap it in the special markers:

<!—- *comment* -->,

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

Language tags: pour identifier le langage du site:

<https://www.w3.org/International/articles/language-tags/>

-----------

Here are a few other sample mailto URLs:

* <mailto:>
* <mailto:nowhere@mozilla.org>
* <mailto:nowhere@mozilla.org,nobody@mozilla.org>
* <mailto:nowhere@mozilla.org?cc=nobody@mozilla.org>
* <mailto:nowhere@mozilla.org?cc=nobody@mozilla.org&subject=This%20is%20the%20subject>
* <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>

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

Description lists use a different wrapper than the other list types — [<dl>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dl); in addition each term is wrapped in a [<dt>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dt) (description term) element, and each description is wrapped in a [<dd>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dd) (description description) element. Let's finish marking up our

--------

Citation

<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>

<p>Hello and welcome to my motivation page. As <a href="http://www.brainyquote.com/quotes/authors/c/confucius.html"><cite>Confucius</cite></a> once said:</p>

<blockquote cite="http://www.brainyquote.com/quotes/authors/c/confucius.html">

<p>It does not matter how slowly you go as long as you do not stop.</p>

</blockquote>

<p>I also love the concept of positive thinking, and <q cite="http://www.affirmationsforpositivethinking.com/index.htm">The Need To Eliminate Negative Self Talk</q> (as mentioned in <a href="http://www.affirmationsforpositivethinking.com/index.htm"><cite>Affirmations for Positive Thinking</cite></a>.)</p>

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

Abréviation

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

<p>I think <abbr title="Reverend">Rev.</abbr> Green did it in the kitchen with the chainsaw.</p>

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

Superscript ou subscript:

<p>My birthday is on the 25<sup>th</sup> of May 2001.</p>

<p>Caffeine's chemical formula is C<sub>8</sub>H<sub>10</sub>N<sub>4</sub>O<sub>2</sub>.</p>

<p>If x<sup>2</sup> is 9, x must equal 3 or -3.</p>

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

Programmation:

* [<code>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/code): For marking up generic pieces of computer code.
* [<pre>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/pre): For marking up blocks of fixed width text, in which the whitespace is retained (generally code blocks.)
* [<var>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/var): For specifically marking up variable names.
* [<kbd>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/kbd): For marking up keyboard (and other types of) input entered into the computer.
* [<samp>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/samp): For marking up the output of a computer program.

-----------

TIME

<!-- Standard simple date -->

<time datetime="2016-01-20">20 January 2016</time>

<!-- Just year and month -->

<time datetime="2016-01">January 2016</time>

<!-- Just month and day -->

<time datetime="01-20">20 January</time>

<!-- Just time, hours and minutes -->

<time datetime="19:30">19:30</time>

<!-- You can do seconds and milliseconds too! -->

<time datetime="19:30:01.856">19:30:01.856</time>

<!-- Date and time -->

<time datetime="2016-01-20T19:30">7.30pm, 20 January 2016</time>

<!-- Date and time with timezone offset-->

<time datetime="2016-01-20T19:30+01:00">7.30pm, 20 January 2016 is 8.30pm in France</time>

<!-- Calling out a specific week number-->

<time datetime="2016-W04">The fourth week of 2016</time>

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

Références pour Text Formatting:

[HTML element reference](https://developer.mozilla.org/en-US/docs/Web/HTML/Element) (the [Inline text semantics](https://developer.mozilla.org/en-US/docs/Web/HTML/Element#Inline_text_semantics) section would be a great place to start.)

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

Structure d'une page WEB:

* **header:** [<header>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/header).
* **navigation bar:** [<nav>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/nav).
* **main content:** [<main>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/main), with various content subsections represented by [<article>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/article), [<section>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/section), and [<div>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/div) elements.
* **sidebar:** [<aside>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/aside); often placed inside [<main>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/main).
* **footer:** [<footer>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/footer).

The  **<div> element** is the generic container for flow content and does not inherently represent anything. Use it to group elements for purposes such as styling (using the [class](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#attr-class) or [id](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#attr-id) attributes), marking a section of a document in a different language (using the [lang](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#attr-lang) attribute), and so on.

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](https://developer.mozilla.org/en-US/docs/Glossary/CSS) or [JavaScript](https://developer.mozilla.org/en-US/docs/Glossary/JavaScript). For cases like these, HTML provides the [<div>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/div) and [<span>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/span) elements. You should use these preferably with a suitable [class](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#attr-class) attribute, to provide some kind of label for them so they can be easily targeted.

HTML element reference:

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

<br> creates a line break in a paragraph; it is the only way to force a rigid structure in a situation where you want a series of fixed short lines, such as in a postal address or a poem

<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 look like a horizontal line.

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

Debugging:

The best strategy is to start by running your HTML page through the [Markup Validation Service](https://validator.w3.org/)

https://validator.w3.org/

Created and maintained by the W3C, the organization that looks after the specifications that define HTML, CSS, and other web technologies. This webpage takes an HTML document as an input, goes through it, and gives you a report to tell you what is wrong with your HTML.

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

IMAGES

<img src="images/dinosaur.jpg">

<img src="https://www.example.com/images/dinosaur.jpg">

The next attribute we'll look at is alt. Its value is supposed to be a textual description of the image, for use in situations where the image cannot be seen/displayed. For example, our above code could be modified like so:

<img src="images/dinosaur.jpg"

alt="The head and torso of a dinosaur skeleton;

it has a large head with long sharp teeth">

Width and height

You can use the width and height attributes, to specify the width and height of your image. You can find your image's width and height in a number of ways. For example on the Mac you can use Cmd + I to get the info display up for the image file. Returning to our example, we could do this:

<img src="images/dinosaur.jpg" alt="The head and torso of a dinosaur skeleton;

it has a large head with long sharp teeth"

width="400"

height="341">

### Image titles

As [with links](https://developer.mozilla.org/en-US/Learn/HTML/Introduction_to_HTML/Creating_hyperlinks#Adding_supporting_information_with_%3Ctitle%3E), you can also add title attributes to images, to provide further supporting information if needed. In our example, we could do this:

<img src="images/dinosaur.jpg"

alt="The head and torso of a dinosaur skeleton;

it has a large head with long sharp teeth"

width="400"

height="341"

title="A T-Rex on display in the Manchester University Museum">

CSS background images

You can also use CSS to embed images into webpages (and JavaScript, but that's another story entirely). The CSS [background-image](https://developer.mozilla.org/en-US/docs/Web/CSS/background-image) property, and the other background-\* properties, are used to control background image placement. For example, to place a background image on every paragraph on a page, you could do this:

p {

background-image: url("images/dinosaur.jpg");

}

The resulting embedded image, is arguably easier to position and control than HTML images. So why bother with HTML images? As hinted to above, CSS background images are for decoration only. If you just want to add something pretty to your page to enhance the visuals, this is fine. Though, such images have no semantic meaning at all. They can't have any text equivalents, are invisible to screen readers, etc. This is HTML images time to shine!

Summing up: if an image has meaning, in terms of your content, you should use an HTML image. If an image is purely decoration, you should use CSS background images.

Video converter:

http://www.mirovideoconverter.com/

Audio converter:

<https://sourceforge.net/projects/audacity/>

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

Embed

<iframe src="https://developer.mozilla.org/en-US/docs/Glossary"

width="100%" height="500" frameborder="0"

allowfullscreen sandbox>

<p> <a href="https://developer.mozilla.org/en-US/docs/Glossary">

Fallback link for browsers that don't support iframes

</a> </p>

</iframe>

[allowfullscreen](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-allowfullscreen)

If set, the <iframe> is able to be placed in fullscreen mode using the [Full Screen API](https://developer.mozilla.org/en-US/docs/Web/Apps/Fundamentals/User_notifications/Full_screen_api) (somewhat beyond scope for this article.)

[frameborder](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-frameborder)

If set to 1, this tells the browser to draw a border between this frame and other frames, which is the default behaviour. 0 removes the border. Using this isn't really recommended any more, as the same effect can be better achieved using [border](https://developer.mozilla.org/en-US/docs/Web/CSS/border): none; in your [CSS](https://developer.mozilla.org/en-US/docs/Glossary/CSS).

[src](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-src)

This attribute, as with [<video>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video)/[<img>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img), contains a path pointing to the URL of the document to be embedded.

[width](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-width) and [height](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-height)

These attributes specify the width and height you want the iframe to be.

**F**allback content

In the same way as other similar elements like [<video>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video), you can include fallback content between the opening and closing <iframe></iframe> tags that will appear if the browser doesn't support the <iframe>. In this case we have included a link to the page instead. It is unlikely that you'll come across any browser that doesn't support <iframe>s these days.

[sandbox](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-sandbox)

This attribute, which works in slightly more modern browsers than the rest of the <iframe> features (e.g. IE 10 and above) requests heightened security settings; we'll say more about this in the next section.

#### Always use the sandbox attribute

You want to give attackers as little power as you can to do bad things on your website, therefore you should give embedded content only the permissions needed for doing its job. Of course, this applies to your own content, too. A container for code where it can be used appropriately — or for testing — but can't cause any harm to the rest of the codebase (either accidental or malicious) is called a [sandbox](https://en.wikipedia.org/wiki/Sandbox_%28computer_security%29).

Unsandboxed content can do way too much (executing JavaScript, submitting forms, popup windows, etc.) By default you should impose all available restrictions by using the sandbox attribute with no parameters, as shown in our previous example.

If absolutely required, you can add permissions back one by one (inside the sandbox="" attribute value) — see the [sandbox](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-sandbox) reference entry for all the available options:

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe#attr-sandbox

One important note is that you should never add both allow-scripts and allow-same-origin to your sandbox attribute — in that case the embedded content could bypass the same origin security policy that stops sites from executing scripts, and use JavaScript to turn off sandboxing altogether.

## The <embed> and <object> elements

The [<embed>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/embed) and [<object>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/object) elements serve a different function to [<iframe>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe) — these elements are general purpose embedding tools for embedding multiple types of external content, which include plugin technologies like Java Applets and Flash, PDF (which can be shown in a browser with a PDF plugin), and even content like videos, SVG and images!

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

Now let's look at an <object> example that embeds a PDF into a page (see the [live example](http://mdn.github.io/learning-area/html/multimedia-and-embedding/other-embedding-technologies/object-pdf.html) and the [source code](https://github.com/mdn/learning-area/blob/gh-pages/html/multimedia-and-embedding/other-embedding-technologies/object-pdf.html)):

<object data="mypdf.pdf" type="application/pdf"

width="800" height="1200" typemustmatch>

<p>You don't have a PDF plugin, but you can <a href="myfile.pdf">download the PDF file.</a></p>

</object>

PDFs were a necessary stepping stone between paper and digital, but they pose many [accessibility challenges](http://webaim.org/techniques/acrobat/acrobat) and can be hard to read on small screens. They do still tend to be popular in some circles, but it is much better to link to them so they can be downloaded or read on a separate page, rather than embedding them in a webpage.

Responsive images

**Note**: In the [<head>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/head) of the document you'll find the line <meta name="viewport" content="width=device-width">: this forces mobile browsers to adopt their real viewport width for loading web pages (some mobile browsers lie about their viewport width, and instead load pages at a larger viewport width then shrink the loaded page down, which is not very helpful for responsive images or design. We'll teach you more about this in a future module.)

**jpg** et **png** compression:

https://tinypng.com/