The Best HTML Examples and HTML5 Examples

```
<head>
  <!--meta-->
  <title></title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0,
  <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon">
  <link rel="icon" href="/favicon.ico" type="image/x-icon">
  <!--css-->
  <link type="text/css" rel="stylesheet" href="css/materialize.min.css"
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.
  <link rel="stylesheet" href="/css/animate.css">
  <link rel="stylesheet" href="css/theme.css">
</head>
<body>
  <!-- banner-->
  <div class="banner">
    <nav class="nav">
      <div class="nav-wrapper">
        <div class="container">
          <a href="#" class="brand-logo hide-on-med-and-up">
        </div>
      </div>
    </nav>
```

HTML provides the structure of websites. Here are some examples of how to use HTML syntax to build websites, including some examples of newer HTML5 features.

The A Href Attribute Example

The <a href> attribute refers to a destination provided by a link. The a (anchor) tag is dead without the <href> attribute. Sometimes in your workflow, you don't want a live link or you won't know the link destination yet. In this case, it's useful to set the href attribute to "#" to create a dead link. The hrefattribute can be used to link to local files or files on the internet.

For instance:

```
<html>
 <head>
  <title>Href Attribute Example</title>
 </head>
 <body>
  <h1>Href Attribute Example</h1>
  <p>
   <a href="https://www.freecodecamp.org/contribute/">The freeCodeCamp Contribution
Page</a> shows you how and where you can contribute to freeCodeCamp's community and
growth.
```

```
   </p>
  </body>
</html>
```

The <a href> attribute is supported by all browsers.

More attributes:

hreflang : Specifies the language of the linked resource. target : Specifies the context in which the linked resource will open. title : Defines the title of a link, which appears to the user as a tooltip.

Examples

```
<a href="#">This is a dead link</a>

<a href="https://www.freecodecamp.org">This is a live link to freeCodeCamp</a>

<a href="https://html.com/attributes/a-href/">more with a href attribute</a>
```

In-page anchors

It’s also possible to set an anchor to certain place of the page. To do this you should first place a tab at location on the page with the <a> tag and the attribute “name” with any keyword description in it, like this:

```
<a name="top"></a>
```

Any description between tags is not required. After that you can place a link leading to this anchor at any palce on same page. To do this you should use tag with necessary attribute “href” with symbol # (sharp) and key-word description of the anchor, like this:

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

Image Links

The <a href="#"> may also be applied to images and other HTML elements.

Example

```
<a href="#">
  <img itemprop="image" style="height: 90px;" src="https://bit.ly/fcc-relaxing-cat" alt="A cute orange cat lying on its back.">
</a>
```

Example

The A Target Example

The <a target> attribute specifies where to open the linked document in an a (anchor) tag.

Examples:

A target attribute with the value of “_blank” opens the linked document in a new window or tab.

```
<a href="https://www.freecodecamp.org/" target="_blank">freeCodeCamp</a>
```

A target attribute with the value of “_self” opens the linked document in the same frame as it was clicked (this is the default and usually does not need to be specified).

```
<a href="https://www.freecodecamp.org/" target="_self">freeCodeCamp</a>
```

```
<a href="https://www.freecodecamp.org/">freeCodeCamp</a>
```

A target attribute with the value of “_parent” opens the linked document in the parent frame.

```
<a href="https://www.freecodecamp.org/" target="_parent">freeCodeCamp</a>
```

A target attribute with the value of “_top” opens the linked document in the full body of the window.

```
<a href="https://www.freecodecamp.org/" target="_top">freeCodeCamp</a>
```

A target attribute with the value of *“framename”* Opens the linked document in a specified named frame.

```
<a href="https://www.freecodecamp.org/" target="framename">freeCodeCamp</a>
```

The Body Background Attribute Example

If you want to add a background image instead of a color, one solution is the <body background> attribute. It specifies a background image for an HTML document.

Syntax:

```
<body background="URL">
```

Attribute:

background - URL for background image

Example:

```
<html>
  <body background="https://assets.digitalocean.com/blog/static/hacktoberfest-is-back/hero.png">
  </body>
</html>
```

body-background attribute is depreciated

the body-background attribute been deprecated in HTML5. The correct way to style the <body> tag is with CSS.

There are several CSS properties used for setting the background of an element. These can be used on to set the background of an entire page.

The Body Bgcolor Attribute Example

The <body bgcolor> attribute assigns a background color for an HTML document.

Syntax:

<body bgcolor="color"> The color value can be either a color name (like, purple) or a hex value (like, #af0000).

To add a background color to a webpage you can use the <body bgcolor="######"> attribute. It specifies a color for the HTML document to display.

For example:

```
<html>
 <head>
  <title>Body bgcolor Attribute example</title>
 </head>
 <body bgcolor="#afafaf">
  <h1>This webpage has colored background.</h1>
 </body>
</html>
```

You can change the color by replacing ###### with a hexadecimal value. For simple colors you can also use the word, such as “red” or “black”.

All major browsers support the <body bgcolor> attribute.

Note:

- HTML 5 does not support the <body bgcolor> attribute. Use CSS for this purpose. How? By using the following code: <body style="background-color: color"> Of course, you can also do it in a separate document instead of an inline method.
- Do not use RGB value in <body bgcolor> attribute because rgb() is for CSS only, that is, it will not work in HTML.

The Div Align Attribute Example

The <div align=""> attribute is used for aligning the text in a div tag to The Left, Right, center or justify.

For instance:

```
<html>
 <head>
  <title>Div Align Attribbute</title>
 </head>
 <body>
  <div align="left">
   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
   labore et dolore magna aliqua.
  </div>
  <div align="right">
   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
   labore et dolore magna aliqua.
  </div>
  <div align="center">
   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
   labore et dolore magna aliqua.
  </div>
  <div align="justify">
   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
   labore et dolore magna aliqua.
  </div>
 </body>
</html>
```

Important!

This attribute is no longer supported in html5. css is the way to go.

The Div Align attribute can be used to horizontally align the contents within a div. In the below example, the text will be centered within the div.

```
<div align="center">
 This Text Will Be Centered
</div>
```

**This attribute is not supported in HTML5 and [CSS Text Align] should be used instead

The Font Color Attribute Example

This attribute is used to set a color to the text enclosed in a <font> tag.

Important:

This attribute is not supported in HTML5. Instead, this freeCodeCamp article specifies a CSS method, which can be used.

Note:

A color can also be specified using a 'hex code' or an 'rgb code', instead of using a name.

Example:

1. Color name attribute

```
<html>
 <body>
  <font color="green">Font color example using color attribute</font>
 </body>
</html>
```

Hex code attribute

```
<html>
 <body>
  <font color="#00FF00">Font color example using color attribute</font>
 </body>
</html>
```

RGB attribute

```
<html>
 <body>
  <font color="rgb(0,255,0)">Font color example using color attribute</font>
 </body>
</html>
```

The Font Size Attribute Example

This attribute specifies the font size as either a numeric or relative value. Numeric values range from 1 to 7with 1 being the smallest and 3 the default. It can also be defined using a relative value, like +2 or -3, which set it relative to the value of the size attribute of the <basefont> element, or relative to 3, the default value, if none does exist.

Syntax:

```
<font size="number">
```

Example:

```
<html>

 <body>

  <font size="6">This is some text!</font>

 </body>

</html>
```

Note : The size attribute of <font> is not supported in HTML5. Use CSS instead.

The Img Align Attribute Example

The align attribute of an image specifies where the image should be aligned according to the surrounding element.

Attribute Values:
right - Align image to the right left - Align image to the left
top - Align image to the top
bottom - Align image to the bottom
middle - Align image to the middle

For example:

```
<!DOCTYPE html>

<html lang="en">

 <head>

  <title>Img Align Attribute</title>

 </head>

 <body>

  <p>

   This is an example. <img src="image.png" alt="Image" align="middle" /> More text right here

   <img src="image.png" alt="Image" width="100" />

  </p>

 </body>

</html>
```

We can also align in right if we want:

```
<p>This is another example<img src="image.png" alt="Image" align="right" /></p>
```

Please note the align attribute is not supported in HTML5, and you should use CSS instead. However, it is still supported by all the major browsers.

The Img Width Attribute

The HTML ‘width’ attribute refers to the width of an image. The value in the quotations is the amount of pixels.

For example, if you already have a link to an image set up via the src attribute you can add the width attribute like so:

```
<!DOCTYPE html>

<html lang="en">

 <head>

  <title>Img Width Attribute</title>

 </head>

 <body>

  <img src="image.png" alt="Image" width="100" />

 </body>

</html>
```

In the code snippet above there is an image tag and the image is set to a width of 100 pixels. width="100"

The Img Src Attribute Example

The <img src> attribute refers to the source of the image you want to display. The img tag will not display an image without the src attribute. However, if you set the source to the location of the image, you can display any image.

There is an image of the freeCodeCamp Logo located at https://avatars0.githubusercontent.com/u/9892522?v=4&s=400

You can set that as the image using the src attribute.

```
<html>

 <head>

  <title>Img Src Attribute Example</title>

 </head>

 <body>

  <img src="https://avatars0.githubusercontent.com/u/9892522?v=4&s=400" />

 </body>

</html>
```

The above code displays like this:

The src attribute is supported by all browsers.

You can also have a locally hosted file as your image.

For example, <img src="images/freeCodeCamp.jpeg> would work if you had a folder called imageswhich had the freeCodeCamp.jpeg inside, as long as the 'images' folder was in the same location as the index.html file.

../files/index.html

..files/images/freeCodeCamp.jpeg

HTML Entity Example

Overview

What are HTML Entities?

HTML entities are characters that are used to replace reserved characters in HTML or for characters that do not appear on your keyboard. Some characters are reserved in HTML. If you use the less than(<) or greater than(>) signs in your text, the browser might mix them up with tags.

What are they used for?

As mentioned about HTML entities are used in order to replace reserved characters that are reserved by HTML.

How do you use them?

A character entity looks similar to this:

```
<!-- format &[entity_name]; -->

<!-- example for a less-than sign (<) -->

&lt;
```

Or

```
<!-- &#[entity_number]; -->

<!-- example for a less-than sign (<) -->

&#60;
```

Reference Guide

This is by no means an exhaustive list but the links below will be able to give you more entities if the ones below do not work for your needs. Happy Coding :bowtie:

Character	Entity Name	Entity Number	Description
		&#32;	Space
!		&#33;	Exclamation mark
”		&#34;	Quotation mark
#		&#35;	Number sign
$		&#36;	Dollar sign
¢	&cent;	&#162;	Cent sign
€	&euro;	&#8364;	Euro sign
£	&pound;	&#163;	GBP sign
¥	&yen;	&#165;	Yen sign
%		&#37;	Percent sign
&	&amp;	&#38;	Ampersand
’		&#39;	Apostrophe
(		&#40;	Opening/Left Parenthesis
)		&#41;	Closing/Right Parenthesis
*		&#42;	Asterisk
+		&#43;	Plus sign
,		&#44;	Comma
-		&#45;	Hyphen

.		&#46;	Period
/		&#47;	Slash
©	&copy;	&#169;	Copyright
®	&reg;	&#174;	Registered Trademark
”	&quot;	&#34;	double quotation mark
>	&gt;	&#62;	Greater-than sign
<	&lt;	&#60;	Less-than sign
•	&bull;	&#8226	Bullet point

HTML Form Example

Basically, forms are used to collect data entered by a user, which are then sent to the server for further processing. They can be used for different kinds of user inputs, such as name, email etc.

Form contains control elements which are wrapped around <form></form> tags, like input, which can have types like:

- text
- email
- password
- checkbox
- radio
- submit
- range
- search
- date
- time
- week
- color
- datalist

Code example:

```
<form>
  <label for="username">Username:</label>
  <input type="text" name="username" id="username" />
  <label for="password">Password:</label>
  <input type="password" name="password" id="password" />
  <input type="radio" name="gender" value="male" />Male<br />
  <input type="radio" name="gender" value="female" />Female<br />
  <input type="radio" name="gender" value="other" />Other
  <input list="Options" />
  <datalist id="Options">
    <option value="Option1"></option>
    <option value="Option2"></option>
    <option value="Option3"></option>
  </datalist>


  <input type="submit" value="Submit" />
  <input type="color" />
  <input type="checkbox" name="correct" value="correct" />Correct
</form>
```

Other elements that form can contain:

- textarea - is a multiline box which is most often used for adding some text eg. comment. Size of textarea is defined by number of rows and columns.
- select - together with <option></option> tag creates drop-down select menu.
- button - The button element can be used to define a clickable button.

MORE INFORMATION ON HTML FORMS.

HTML Forms are required when you want to collect some data from the site visitor. For example, during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML <form> tag is used to create an HTML form and it has following syntax −

<form action="Script URL" method="GET|POST">form elements like input, textarea etc.</form>

If the form method is not defined then it will default to “GET”.

The form tag can also have an attribute named “target” which specifies where the link will open. It can open in the browser tab, a frame, or in the current window.

The action attribute defines the action to be performed when the form is submitted. Normally, the form data is sent to a web page at the Script URL when the user clicks on the submit button. If the action attribute is omitted, the action is set to the current page.

HTML5 Audio Example

Before HTML5, audio files had to be played in a browser using a plug-in like Adobe Flash. The HTML

The following code snippet adds an audio file with the filename tutorial.ogg or tutorial.mp3. Theelement indicates alternative audio files which the browser may choose from. The browser will utilize the first recognized format.

Example 1

```
<audio controls>

  <source src="tutorial.ogg" type="audio/ogg" />

  <source src="tutorial.mp3" type="audio/mpeg" />

  Your browser does not support the audio element.

</audio>
```

Example 2

```
<audio src="https://s3.amazonaws.com/freecodecamp/simonSound1.mp3" controls loop autoplay></audio>
```

The controls attribute includes audio controls like play, pause, and volume. If you don't use this attribute, then no controls will be shown.

The <source> element enables you to indicate alternative audio files which the browser may choose from. The browser will utilize the first recognize format. The text between the <audio> and </audio> tags may be shown in browser that does not support the HTML5 <audio> element.

The autoplay attribute will automatically play your audio file in the background. It is considered better practice to let visitors choose to play audio.

The preload attribute indicates what the browser should do if the player is not set to autoplay.

The loop attribute will play your audio file in a continous loop if mentioned

Since this is html5, some browser do not support it. You can check it at https://caniuse.com/#search=audio

HTML5 Semantic Elements Example

Semantic HTML elements clearly describe it's meaning in a human and machine readable way. Elements such as <header>, <footer> and <article> are all considered semantic because they accurately describe the purpose of the element and the type of content that is inside them.

A Quick History

HTML was originally created as a markup language to describe documents on the early internet. As the internet grew and was adopted by more people, it’s needs changed. Where the internet was originally inteded for sharing scientific documents, now people wanted to share other things as well. Very quickly, people started wanting to make the web look nicer. Because the web was not initially built to be designed, programmers used different hacks to get things laid out in different ways. Rather than using the <table></table> to describe information using a table, programmers would use them to position other elements on a page. As the use of visually designed layouts progressed, programmers started to use a generic “non-semantic” tag like <div>. They would often give these elements a class or id attribute to describe their purpose. For example, instead of <header> this was often written as <div class="header">. As HTML5 is still relatively new, this use of non-semantic elements is still very common on websites today.

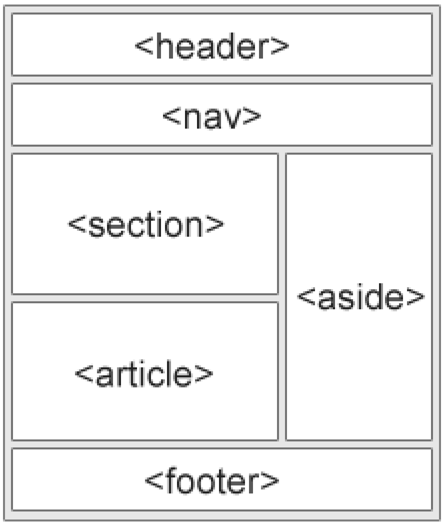
List of new semantic elements

The semantic elements added in HTML5 are:

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>

Elements such as <header>, <nav>, <section>, <article>, <aside>, and <footer> act more or less like <div> elements. They group other elements together into page sections. However where a <div> tag could contain any type of information, it is easy to identify what sort of information would go in a semantic <header> region.

An example of semantic element layout by w3schools



Benefits of semantic elements

To look at the benefits of semantic elements, here are two pieces of HTML code. This first block of code uses semantic elements:

```
<header></header>
<section>
 <article>
  <figure>
   <img />
   <figcaption></figcaption>
  </figure>
 </article>
</section>
<footer></footer>
```

Whilst this second block of code uses non-semantic elements:

```
<div id="header"></div>
<div class="section">
 <div class="article">
  <div class="figure">
   <img />
   <div class="figcaption"></div>
  </div>
 </div>
```

```
</div>

<div id="footer"></div>
```

First, it is much **easier to read**. This is probably the first thing you will notice when looking at the first block of code using semantic elements. This is a small example, but as a programmer you can be reading through hundreds or thousands of lines of code. The easier it is to read and understand that code, the easier it makes your job.

It has **greater accessibility**. You are not the only one that finds semantic elements easier to understand. Search engines and assistive technologies (like screen readers for users with a sight impairment) are also able to better understand the context and content of your website, meaning a better experience for your users.

Overall, semantic elements also lead to more **consistent code**. When creating a header using non-semantic elements, different programmers might write this as <div class="header">, <div id="header">, <div class="head">, or simply <div>. There are so many ways that you can create a header element, and they all depend on the personal preference of the programmer. By creating a standard semantic element, it makes it easier for everyone.

Since October 2014, HTML4 got upgraded to HTML5, along with some new “semantic” elements. To this day, some of us might still be confused as to why so many different elements that doesn’t seem to show any major changes.

<section> **and** <article>

“What’s the difference?”, you may ask. Both these elements are used for sectioning a content, and yes, they can definitely be used interchangeably. It’s a matter of in which situation. HTML4 offered only one type of container element, which is <div>. While this is still used in HTML5, HTML5 provided us with <section>and <article> in a way to replace <div>.

The <section> and <article> elements are conceptually similar and interchangeable. To decide which of these you should choose, take note of the following:

1. An article is intended to be independently distributable or reusable.
2. A section is a thematic grouping of content.

```
<section>

 <p>Top Stories</p>

 <section>

  <p>News</p>

  <article>Story 1</article>

  <article>Story 2</article>

  <article>Story 3</article>

 </section>

 <section>

  <p>Sport</p>
```

```
    <article>Story 1</article>
    <article>Story 2</article>
    <article>Story 3</article>
  </section>
</section>
```

<header> **and** <hgroup>

The <header> element is generally found at the top of a document, a section, or an article and usually contains the main heading and some navigation and search tools.

```
<header>
  <h1>Company A</h1>
  <ul>
    <li><a href="/home">Home</a></li>
    <li><a href="/about">About</a></li>
    <li><a href="/contact">Contact us</a></li>
  </ul>
  <form target="/search">
    <input name="q" type="search" />
    <input type="submit" />
  </form>
</header>
```

The <hgroup> element should be used where you want a main heading with one or more subheadings.

```
<hgroup>
  <h1>Heading 1</h1>
  <h2>Subheading 1</h2>
  <h2>Subheading 2</h2>
</hgroup>
```

REMEMBER, that the <header> element can contain any content, but the <hgroup> element can only contain other headers, that is <h1> to <h6> and including <hgroup>.

<aside>

The <aside> element is intended for content that is not part of the flow of the text in which it appears, however still related in some way. This of <aside> as a sidebar to your main content.

```
<aside>
 <p>
  This is a sidebar, for example a terminology definition or a short background to a historical
  figure.
 </p>
</aside>
```

Before HTML5, our menus were created with <ul>’s and <li>’s. Now, together with these, we can separate our menu items with a <nav>, for navigation between your pages. You can have any number of <nav> elements on a page, for example, its common to have global navigation across the top (in the <header>) and local navigation in a sidebar (in an <aside> element).

```
<nav>
 <ul>
  <li><a href="/home">Home</a></li>
  <li><a href="/about">About</a></li>
  <li><a href="/contact">Contact us</a></li>
 </ul>
</nav>
```

<footer>

If there is a <header> there must be a <footer>. A <footer> is generally found at the bottom of a document, a section, or an article. Just like the <header> the content is generally metainformation, such as author details, legal information, and/or links to related information. It is also valid to include <section>elements within a footer.

```
<footer>&copy;Company A</footer>
```

<small>

The <small> element often appears within a <footer> or <aside> element which would usually contain copyright information or legal disclaimers, and other such fine print. However, this is not intended to make the text smaller. It is just describing its content, not prescribing presentation.

```
<footer><small>&copy;Company A</small> Date</footer>
```

<time>

The <time> element allows an unambiguous ISO 8601 date to be attached to a human-readable version of that date.

```
<time datetime="2017-10-31T11:21:00+02:00">Tuesday, 31 October 2017</time>
```

Why bother with <time>? While humans can read time that can disambiguate through context in the normal way, the computers can read the ISO 8601 date and see the date, time, and the time zone.

<figure> **and** <figcaption>

<figure> is for wrapping your image content around it, and <figcaption> is to caption your image.

```
<figure>
  <img src="https://en.wikipedia.org/wiki/File:Shadow_of_Mordor_cover_art.jpg" alt="Shadow of Mordor" />
  <figcaption>Cover art for Middle-earth: Shadow of Mordor</figcaption>
</figure>
```

HTML5 Video Example

Before HTML5, in order to have a video play in a webpage you would need to use a plugin, like Adobe Flash Player. With the introduction of HTML5, you can now place it directly into the page itself. The HTML

To embed video file into web page, just add this code snippet and change the src of audio file.

```
<video controls>
  <source src="tutorial.ogg" type="video /ogg" />
  <source src="tutorial.mp4" type="video /mpeg" />
  Your browser does not support the video element. Kindly,update it to latest version.
</video>
```

The controls attribute includes video controls, similar to play, pause, and volume.

This feature is supported by all modern/updated browsers. However, not all support the same video file format. My recommendation for a wide range of compatibilty is MP4, as it is the most widely accepted format. There are also two other formats (WebM and Ogg) that are supported in Chrome, Firefox, and Opera.

The element enables you to indicate alternative video files which the browser may choose from. The browser will utilize the first recognize format. In HTML5, there are 3 supported video formats: MP4, WebM, and Ogg.

The text between the tags will only be displayed in browsers that do not support the

There are several different elements of the video tag, many of these explanations are based on Mozilla's web docs (linked below). There are even more if you click the link at the bottom.

autoplay

"autoplay" can be set to either true or false. You set it to true by adding it into the tag, if it is not present in the tag it is set to false. If set to true, the video will begin playing as soon as enough of the video has buffered for it to be able to play. Many people find autoplaying videos as disruptive or annoying so use this feature sparingly. Also note, that some mobile browsers, such as Safari for iOS, ignore this attribute.

```
<video autoplay>
  <source src="video.mp4" type="video/mp4" />
</video>
```

poster

The “poster” attribute is the image that shows on the video until the user clicks to play it.

```
<video poster="poster.png">
  <source src="video.mp4" type="video/mp4" />
</video>
```

controls

The “controls” attribute can be set to true or false and will handle whether controls such as the play/pause button or volume slider appear. You set it to true by adding it into the tag, if it is not present in the tag it is set to false.

```
<video controls>
  <source src="video.mp4" type="video/mp4" />
</video>
```

There are many more attributes that can be added that are optional to customize the videoplayer in the page. To learn more, click on the links below.

HTML5 Web Storage Example

Web storage allows web applications to store up to 5MB of information in browser storage per origin (per domain and protocol).

Types of Web Storage

There are two objects for storing data on the client:

window.localStorage: stores data with no expiration date and lives until removed.

```
// Store Item
localStorage.setItem("foo", "bar");

// Get Item
localStorage.getItem("foo"); //returns "bar"
```

window.sessionStorage: stores data for one session, where data is lost when the browser / browser tab is closed.

```
// Store Item
```

```
sessionStorage.setItem("foo", "bar");


// Get Item

sessionStorage.getItem("foo"); //returns "bar"
```

Since the current implementation only supports string-to-string mappings, you need to serialize and de-serialize other data structures.

You can do so using JSON.stringify() and JSON.parse().

For e.g. for the given JSON

```
var jsonObject = { 'one': 1, 'two': 2, 'three': 3 };
```

We first convert the JSON object to string and save in the local storage:

```
localStorage.setItem('jsonObjectString', JSON.stringify(jsonObject));
```

To get the JSON object from the string stored in local storage:

```
var jsonObject = JSON.parse(localStorage.getItem('jsonObjectString'));
```

Mailto Links Example

A mailto link is a kind of hyperlink (<a href=""></a>) with special parameters that lets you specify additional recipients, a subject line, and/or a body text.

The basic syntax with a recipient is:

```
<a href="mailto:friend@something.com">Some text</a>
```

More customization!

Adding a subject to that mail:

If you want to add a specific subject to that mail, be careful to add %20 or + everywhere there's a space in the subject line. An easy way to ensure that it is properly formatted is to use a URL Decoder / Encoder.

Adding body text:

Similarly, you can add a specific message in the body portion of the email: Again, spaces have to be replaced by %20 or +. After the subject paramater, any additional parameter must be preceded by &

Example: Say you want users to send an email to their friends about their progress at Free Code Camp:

Address: empty

Subject: Great news

Body: I am becoming a developer

Your html link now:

```
<a href="mailto:?subject=Great%20news&body=I%20am%20becoming%20a%20developer">Send mail!</a>
```

Here, we’ve left mailto empty (mailto:?). This will open the user’s email client and the user will add the recipient address themselves.

Adding more recipients:

In the same manner, you can add CC and bcc parameters. Separate each address by a comma!

Additional parameters must be preceded by &.

```
<a href="mailto:firstfriend@something.com?subject=Great%20news&cc=secondfriend@something.com,thirdfriend@something.com&bcc=fourthfriend@something.com">Send mail!</a>
```

Thank you for using this HTML reference. Happy coding!