

HTML and CSS

By Fred Bolder

Foreword

This document contains some HTML and CSS examples.

Animation

animation-delay: 500ms;

animation-timing-function: ease; (other: ease-in, ease-out, ease-in-out, linear, step-start, step-end)

animation-timing-function: steps(5, end);

animation-timing-function: cubic-bezier(0.1, -0.6, 0.2, 0);

animation-play-state: running; (other: paused)

animation-direction: normal; (other: reverse, alternate, alternate-reverse)

Example

```
@keyframes my-animation {  
  0% {  
    background-color: white;  
  }  
  50% {  
    background-color: blue;  
  }  
  100% {  
    background-color: white;  
  }  
}
```

```
.box {  
  width: 100px;  
  height: 100px;  
  background-color: white;  
  border: 1px solid blue;  
  animation-duration: 1000ms;  
  animation-timing-function: ease-in-out;  
  animation-iteration-count: infinite;  
}
```

```
.box:hover {  
  animation-name: my-animation;  
}
```

Example No animation

```
@media (prefers-reduced-motion) {  
  animation-play-state: paused;  
}
```

Links

<https://animista.net/>

<https://cubic-bezier.com/>

Background color

CSS

background-color: red;
background-color: #ff0000; (red in hex)
background-color: rgb(255, 0, 0); (red in rgb)
background-color: rgba(255, 0, 0, 0.5); (50% transparent)

Background image

CSS

background-image: url("image.jpg");
background-repeat: repeat; (other: no-repeat, space, round)
background-size: auto; (other cover, contain)
background-position: bottom;
background-image: linear-gradient(45deg, #000060, #0000c0);

Links

<https://cssgradient.io/>

Bold

HTML

text

CSS

font-weight: bold;

Border

CSS

border-width: 1px;
border-style: solid;
border-color: black;

Shorthand for above settings

border: 1px solid black;

Tip

border-bottom can also be used to underscore

Center element

CSS

width: 50%;
margin: auto;

Color

CSS

background-color: blue;
background-color: #0000ff; (blue in hex)
background-color: #2af; (same as #22aaff)
background-color: rgb(0, 0, 255); (blue in rgb)
background-color: rgba(0, 0, 255, 0.2); (80% transparent)

Comment

HTML

<!-- text -->

CSS

/ text */*

CSS

Cascading Style Sheets

<https://htmlcheatsheet.com/css/>

Flexbox container

display: flex;

flex-direction: row; (other: column, row-reverse, column-reverse)

flex-wrap: nowrap; (other: wrap, wrap-reverse)
wrap = elements that don't fit go to the next line

justify-content: flex-start;

This property distributes elements along the main axis.

123				flex-start
			123	flex-end
		123		center
1		2	3	space-between
	1	2	3	space-around
	1	2	3	space-evenly

align-items: stretch; (other: flex-start, flex-end, center, baseline)

This property aligns elements on the cross axis.

align-content: flex-start; (other: flex-end, center, stretch, space-between, space-around)

This property aligns multi-line content on the cross axis.

The gap, row-gap and column-gap properties control the space between flex items.

<https://blogs.cril.com/css-layout-introduction-flexbox-userInterface/>

Flexbox item

A flexbox item is a direct child in a flexbox container.

align-self: auto; (other flex-start, flex-end, center, baseline, stretch)

flex-basis: 20%

flex-grow: 1.5; (positive factor, 0 = no grow)

flex-shrink: 2; (positive factor 0 = no shrink)

order: 0; (integer, can also be negative)

Higher is closer to the end and lower is closer to the start. Elements with the same order value are sorted in the main direction depending on the html order.

Grid container

The grid is invisible and has no background colors and borders.

```
display: grid;
```

```
grid-template-columns: 1fr 2fr; (width c2 = 2 x width c1)
```

```
grid-template-columns: 100px repeat(3, 1fr);
```

```
grid-template-columns: 1fr minmax(200px, 1fr) 1fr;
```

```
grid-template-rows: 1fr 1fr 1fr;
```

You don't need to specify both grid-template-columns and grid-template-rows.

```
justify-items: center; (always horizontal)
```

```
align-items: start; (always vertical)
```

```
gap: 1rem; (distance between the cells)
```

```
column-gap: 1.5rem; (distance between the columns)
```

```
row-gap: 2.5rem; (distance between the rows)
```

<https://gridbyexample.com/>

<https://cssgrid.io/>

Grid item

A grid item is a direct child in a grid container.

Here is a grid. The negative numbers start from the end, so -1 is always the end.

	-4	-3	-2	-1
-4	1	2	3	4
			2	
-3	2	3	3	3
		3	3	3
-2	3			1
				1
-1	4			

For a grid item you can define the area as follows:

Item 1

```
grid-column-start: 3;  
grid-row-start: 3;  
grid-column-end: 4;  
grid-row-end: 4;
```

In this case, you don't need to specify the ends, because the default span is 1.

Item 2

```
grid-column-start: 3;  
grid-row-start: 1;
```

Item 3

This is the whole second row.

```
grid-column-start: 1;  
grid-row-start: 2;  
grid-column-end: 4;
```

Since it is the whole row, you can also define the following:

```
grid-column-start: 1;  
grid-row-start: 2;  
grid-column-end: -1;
```

You can also use shorthands.

```
grid-column: 1 / 4;  
grid-row: 2 / 3;
```

or

```
grid-column: 1 / span 3;  
grid-row: 2 / span 1;
```

Grid areas must be rectangular and can overlap.

You can name a grid item to use it in the grid container property `grid-template-areas`.

```
grid-area: nav;
```

HTML

Hypertext Markup Language

Image

```

```

Styling in CSS

object-fit: cover; (keeps ratio, but crops)

object-fit: contain; (keeps ratio, but adds space)

Internal link

```
<a href="#about-me">About me</a>
```

```
<h2 id="about-me">About me</h2>
```

Link stylesheet file

```
<link rel="stylesheet" href="/main.css" />
```

Semantic elements

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

Units

px = pixels

% = percentage

em = relative to the font size of the element

rem = relative to the font size of the root element

vw = percentage of viewport width

vh = percentage of viewport height

Uppercase

CSS

```
text-transform: uppercase;
```