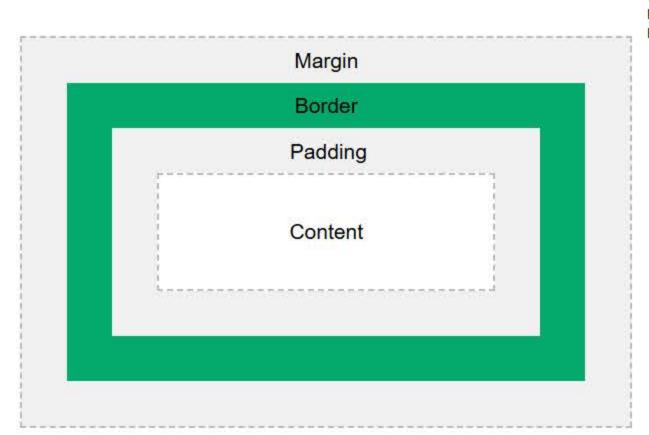
CSS Box Model



Property

Content - The content of the box, where text and images appear Padding - Clears an area around the content. The padding is transparent Border - A border that goes around the padding and content Margin - Clears an area outside the border. The margin is transparent

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
```

CSS has a lot of properties for formatting text:

aligment decoration transformation spacing shadow

```
p {
  color: red;
  text-align: center;
```

p is a selector in CSS (it points to the HTML element you want to style:). color is a property, and red is the property value

text-align is a property, and center is the property value

simple selectors

CSS Syntax CSS Selectors

Simple selectors select elements based on name, id, class relations Combinator selectors select elements based on a specific relationship state Pseudo-class selectors select elements based on a certain state part Pseudo-elements selectors select and style a part of an element value Attribute selectors select elements based on an attribute or value

Selector Declaration Declaration {color:blue; font-size:12px;}

Property Value

CSS selectors are used to "find" (or select) the HTML elements you want to style.

Selector	Example	Example description
#id	#firstname	Selects the element with id="firstname"
.class	.intro	Selects all elements with class="intro"
*	*	Selects all elements
element	р	Selects all elements
element,element,	div, p	Selects all <div> elements and all elements</div>

CSS Comments

color: /*red*/blue;

Hello World!

color: red;

p {

p {

/* This is a single-line comment */

<!-- These paragraphs will be red -->

This paragraph is styled with CSS.

CSS comments are not shown in the output.

color: red; /* Set text color to red */

CSS id Selector

#para1 {

```
text-align: center;
   color: red;
Hello World!
This paragraph is not affected by the style.
The id of an element is unique within a page, so the id
```

How To Add CSS

External CSS Internal CSS Inline CSS

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules:

Inline style (inside an HTML element)

selector is used to select one unique element!

External and internal style sheets (in the head section) Browser default

internal vs. external

<!DOCTYPE html>

```
<html>
<head>
<style>
h1 {
  color: orange;
</style>
<link rel="stylesheet" type="text/css"</pre>
href="mystyle.css">
</head>
<body>
<h1>This is a heading</h1>
The style of this document is a combination of an
external stylesheet, and internal style
</body>
</html>
```

CSS class Selector

```
p.center {
.center {
                          text-align: center;
 text-align: center;
                          color: red;
 color: red;
```

The class selector selects HTML elements with a specific class attribute; with a specific class, write a period (.) character,

```
<html>
<head>
</head>
<body>
<h1>This is a heading</h1>
This is a paragraph.
</body>
</html>
mystyle.css
body {
  background-color: lightblue;
h1 {
  color: navy;
  margin-left: 20px;
```

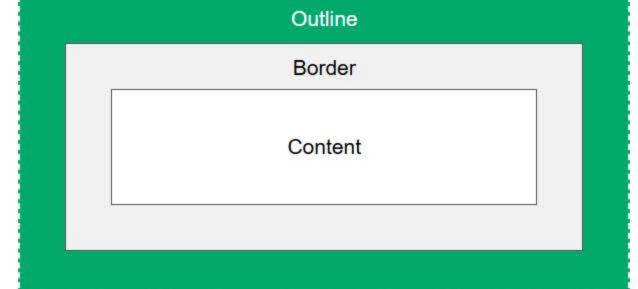
external style sheet, the <h1> elements will be "navy", not orange, If the internal style is defined elements will be "orange"

Value

```
followed by the class name.
external
<!DOCTYPE html>
<link rel="stylesheet" href="mystyle.css">
```

If the internal style is defined before the link to the after the link to the external style sheet, the <h1>

CSS Outline



An outline is a line that is drawn around elements, OUTSIDE the borders, to make the element "stand out".

dotted - Defines a dotted outline dashed - Defines a dashed outline solid - Defines a solid outline double - Defines a double outline groove - Defines a 3D grooved outline ridge - Defines a 3D ridged outline inset - Defines a 3D inset outline outset - Defines a 3D outset outline none - Defines no outline hidden - Defines a hidden outline

Generic Font Family

Serif

Sans-serif

Monospace

Cursive

CSS Universal Selector

```
text-align: center;
color: blue;
```

internal

<html>

<head>

<style>

body {

</style>

</head>

<body>

</body>

</html>

<!DOCTYPE html>

color: maroon;

<h1>This is a heading</h1>

This is a paragraph.

The universal selector (*) selects all HTML elements on the page.

inline

```
text-align: center;
color: red;
```

h1, h2, p {

CSS Grouping Selector

<!DOCTYPE html> <html> <body> <h1 style="color:blue;textalign:center; ">This is a heading</h1> This is a paragraph background-color: linen; </body> </html> In this example the element will be styled margin-left: 40px; according to class="center" and to class="large":

This paragrap <style> p.center { text-align: center; color: red; p.large { font-size: 300%; </style>

A dotted border. A dashed border. A solid border.

......

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

...... A mixed border.

Examples of Font Names

Times New Roman

Georgia

Arial

Garamond

Verdana

Helvetica

Monaco

Courier New

Lucida Console

Brush Script MT

Copperplate

Papyrus

Lucida Handwriting

CSS Fonts

Serif fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance. Sans-serif fonts have clean lines (no small strokes attached). They create a modern and minimalistic look. Monospace fonts - here all the letters have the same fixed width. They create a mechanical look. Cursive fonts imitate human handwriting. Fantasy fonts are decorative/playful fonts.

Tip: The font-family property should hold several font names as a "fallback" system, to ensure maximum compatibility between browsers/operating systems. Start with the font you want, and end with a generic family (to let the browser pick a similar font in the generic family, if no other fonts are available). The font names should be separated with a comma.

Font Awesome Icons

```
<!DOCTYPE html>
<html>
<head>
<script src="https://kit.fontawesome.com/a076d05399.js"</pre>
```

<pre>crossorigin="anonymous"></pre>	ipt>
<body></body>	Fantasy
<pre><i class="fas fa-cloud"></i> </pre>	<pre>Styled Font Awesome icons (size and color): <i class="fas fa-cloud" style="font-size:24px;"></i> <i class="fas fa-cloud" style="font-size:36px;"></i> <i class="fas fa-cloud" style="font-size:48px;color:red;"></i></pre>
	<pre><i class="fas fa-cloud" style="font-size:60px;color:lightblue;"></i></pre> //

Styling Links

The four links states are:

a:link - a normal, unvisited link a:visited - a link the user has visited a:hover - a link when the user mouses over it a:active - a link the moment it is clicked /* unvisited link */ a:link { color: red; /* visited link */ a:visited { color: green; /* mouse over link */ a:hover { color: hotpink; /* selected link */ a:active {

color: blue;

CSS Lists

unordered lists () - the list items are marked with bullets ordered lists () - the list items are marked with numbers or letters

Set different list item markers for ordered lists Set different list item markers for unordered lists Set an image as the list item marker Add background colors to lists and list items

The list-style property is a shorthand property. It is used to set all the list properties in one declaration:

```
ul {
 list-style: square inside url("sqpurple.gif");
```

CSS Tables

Table properties: table borders table size table, th, td { table alignment border: 1px solid; table style table responsive

CSS Layout - The display Property

The display property is the most important CSS property for controlling layout. The default display value for most elements is block or inline.

Block-level Elements

A block-level element ALWAYS starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

Note: Setting the display property of an

element only changes how the element is

displayed, NOT what kind of element it is. So,

allowed to have other block elements inside it.

an inline element with display: block; is not

The <div> element is a block-level element.

Examples of block-level elements:

- <div> <h1> - <h6>
- <
- <form>
- <header>
- <footer>
- <section>

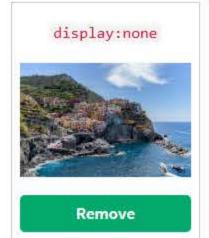
Inline Elements

An inline element DOES NOT start on a new line and only takes up as much width as necessary.

This is an inline element inside a paragraph.

Examples of inline elements:

-
- <a>
-







Property	Description
display	Specifies how an element should be displayed
visibility	Specifies whether or not an element should be visible

ul.a { list-style-type: circle; ul.b { list-style-type: square; ol.c { list-style-type: upper-roman; ol.d {

list-style-type: lower-alpha;

Property	Description
<u>list-style</u>	Sets all the properties for a list in one declaration
<u>list-style-image</u>	Specifies an image as the list-item marker
list-style-position	Specifies the position of the list-item markers (bullet points)
<u>list-style-type</u>	Specifies the type of list-item marker

ul.a { The list-style-image property specifies an image as the list item list-style-position: outside; ul { list-style-image: url('sqpurple.gif'); ul.b { list-style-position: inside;

display property

list property

splay property	
Value	Description
inline	Displays an element as an inline element
block	Displays an element as a block element
contents	Makes the container disappear, making the child elements children of the element the next level up in the DOM
flex	Displays an element as a block-level flex container
grid	Displays an element as a block-level grid container
inline-block	Displays an element as an inline-level block container. The element itself is formatted as an inline element, but you can apply height and width values
inline-flex	Displays an element as an inline-level flex container
inline-grid	Displays an element as an inline-level grid container
inline-table	The element is displayed as an inline-level table
list-item	Let the element behave like a element
run-in	Displays an element as either block or inline, depending on context
table	Let the element behave like a element
table-caption	Let the element behave like a <caption> element</caption>
table-column- group	Let the element behave like a <colgroup> element</colgroup>
table-header- group	Let the element behave like a <thead> element</thead>
table-footer- group	Let the element behave like a <tfoot> element</tfoot>
table-row-group	Let the element behave like a element
table-cell	Let the element behave like a element
table-column	Let the element behave like a <col/> element
table-row	Let the element behave like a > element
none	The element is completely removed
initial	Sets this property to its default value

A common example is making inline elements for horizontal menus: <!DOCTYPE html>

```
<html>
<head>
<style>
li {
 display: inline;
</style>
</head>
<body>
Display a list of links as a
horizontal menu:
<a href="/html/default.asp"</a>
target="_blank">HTML</a>
  <a href="/css/default.asp"</p>
target=" blank">CSS</a>
  <a href="/js/default.asp"</p>
target="_blank">JavaScript</a>
```

Display a list of links as a horizontal menu:

HTML CSS JavaScript

</body>

</html>

<html>

<head>

<style>

The element will still take up the same space as before, it will be hidden, but still affect the layout: <!DOCTYPE html>

```
h1.hidden {
  visibility: hidden;
</style>
</head>
<body>
<h1>This is a visible heading</h1>
<h1 class="hidden">This is a hidden
heading</h1>
Notice that the hidden heading
still takes up space.
```

</body> </html>

div.relative {

div.absolute {

width: 400px;

height: 200px;

position: relative;

position: absolute;

This is a visible heading

Notice that the hidden heading still takes up

Using width, max-width and margin: auto;

As mentioned in the previous chapter; a block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Setting the width of a block-level element will prevent it from stretching out to the edges of its container. Then, you can set the margins to auto, to horizontally center the element within its container. The element will take up the specified width, and the remaining space will be split equally between the two margins:

> This <div> element has a width of 500px, and margin set to auto.

Note: The problem with the <div> above occurs when the browser window is smaller than the width of the element. The browser then adds a horizontal scrollbar to the page.

Using max-width instead, in this situation, will improve the browser's handling of small

windows. This is important when making a site usable on small devices:

This <div> element has a max-width of 500px, and margin set to auto.

CSS Layout - The position

There are five different position values: static relative fixed absolute sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work unierently depending on the position value.

div.sticky { position: sticky; top: 0; background-color: green; border: 2px solid #4CAF50; div.static { position: static; border: 3px solid #73AD21;

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page.

An element with position: sticky; is positioned based on the user's scroll position. A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed). center {

<style> position: absolute; width: 100%; .container top: 50%; height: auto; position: relative; width: 100%; opacity: 0.3; text-align: center; font-size: 18px; </style>

<div class="container"> <img src="img 5terre_wide.jpg" alt="Cinque Terre" width="1000"</pre> <div class="center">Centered</div> </div>

auto. Note: The problem with the <div> above occurs when the browser window is smaller than the width of the element. The browser then adds a horizontal scrollbar to the page.

Inherits this property from its parent element

Using max-width instead, in this situation, will improve the browser's handling of small windows. This is important when making a site usable on small devices:

This <div> element has a width of 500px, and magin set to

This <div> element has a max-width of 500px, and margin set to auto.

This <div> element has position: relative;

Here is the CSS that is used: div.relative { position: relative; left: 30px; border: 3px solid #73AD21;

inherit

Setting the top, right, bottom, and left properties of a relativelypositioned element will cause it to be adjusted away from its normal position.

An element with position: fixed; is same place even if the page is the element, it does not leave a gap in the page where it would normally have been located.

div.fixed { position: fixed; bottom: 0; right: 0; width: 300px;

positioned relative to the viewport, which means it always stays in the scrolled. The top, right, bottom, and left properties are used to position

border: 3px solid #73AD21;

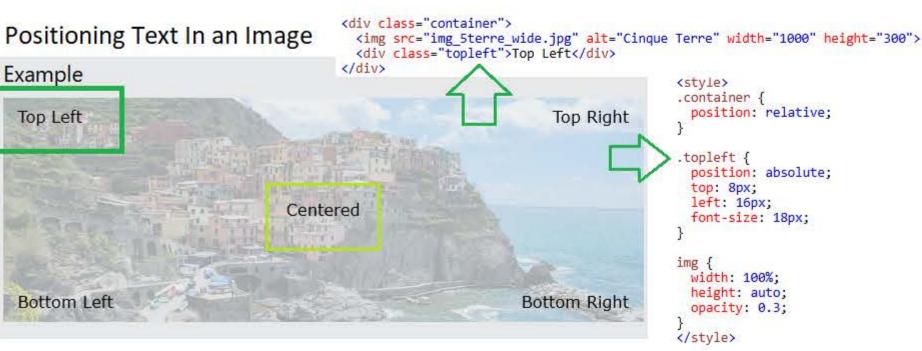
top: 80px; right: 0; width: 200px; height: 100px; border: 3px solid #73AD21; An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

border: 3px solid #73AD21;

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling. Note: Absolute positioned elements are removed from the

normal flow, and can overlap elements. This <div> element has position: relative;

This <div> element has position: absolute;



```
z-index: -1;
Note: z-index only works on positioned elements
(position: absolute, position: relative, position: fixed, or
position: sticky) and flex items (elements that are direct
children of display: flex elements).
<!DOCTYPE html>
<html>
<head>
<style>
.container {
 position: relative;
.black-box {
  position: relative;
  z-index: 1;
  border: 2px solid black;
  height: 100px;
  margin: 30px;
.gray-box {
  position: absolute;
  z-index: 3; /* gray box will be above both green and black
  background: lightgray;
  height: 60px;
  width: 70%;
  left: 50px;
 top: 50px;
.green-box {
  position: absolute;
  z-index: 2; /* green box will be above black box */
  background: lightgreen;
  width: 35%;
  left: 270px;
  top: -15px;
  height: 100px;
</style>
</head>
<h1>Z-index Example</h1>
An element with greater stack order is always above an
element with a lower stack order.
<div class="container">
  <div class="black-box">Black box (z-index: 1)</div>
  <div class="gray-box">Gray box (z-index: 3)</div>
```

CSS Layout - The z-index Property

Which element should be placed in front of, or behind,

position: absolute;

left: 0px;

top: 0px;

The z-index property specifies the stack order of an element.

Z-index Example

</div>

</body>

An element with greater stack order is always above an element with a lower stack order.

<div class="green-box">Green box (z-index: 2)</div>

```
Green box (z-index: 2)
Black box (z-index: 1)
  Gray box (z-index: 3)
```

CSS Combinators

A combinator is something that explains the relationship between the selectors.

Descendant combinator (space) Child combinator (>)

Next sibling combinator (+)

Subsequent-sibling combinator (~)

CSS Units

absolute lengths

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
рс	picas (1pc = 12 pt)
relative	lengths

next sibling combinator +

```
div > p {
  background-color: yellow;
```

The next sibling combinator is used to select an element that is directly after another specific

Sibling elements must have the same parent element, and "adjacent" means "immediately following".

relativ	relative lengths		
Unit	Description		
em	Relative to the font-size of the element (2em means 2 times the size of the current font)		
ex	Relative to the x-height of the current font (rarely used)		
ch	Relative to width of the "0" (zero)		
rem	Relative to font-size of the root element		
vw	Relative to 1% of the width of the viewport*		
vh	Relative to 1% of the height of the viewport*		
vmin	Relative to 1% of viewport's* smaller dimension		
vmax	Relative to 1% of viewport's* larger dimension		
%	Relative to the parent element		

CSS Counters

body {

CSS counters are like "variables". The variable values can be incremented by CSS rules (which will track how many times they are used).

```
counter-reset: section;
h2::before {
  counter-increment: section;
  content: "Section " counter(section) ": ";
```

CSS Layout - Overflow

The CSS overflow property controls what happens to content that is too big to fit into an area. The overflow property has the following values:

visible - Default. The overflow is not clipped. The content renders outside the element's box

hidden - The overflow is clipped, and the rest of the content will be invisible scroll - The overflow is clipped, and a scrollbar is added to see

the rest of the content auto - Similar to scroll, but it adds scrollbars only when necessary

CSS Layout - float and clear

The CSS float property specifies how an element should float. The CSS clear property specifies what elements can float beside the cleared element and on which side.

left - The element floats to the left of its container right - The element floats to the right of its container

none - The element does not float (will be displayed just where it

occurs in the text). This is default inherit - The element inherits the float value of its parent

```
<style>
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        in the sample, it camposed the following bland represent and the following to properly with a representation to the longer than the control of the longer than the properly of the control of the longer than 
img {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     Livers operated by an environmental and potential of
Phaselics imperior, and also diction interform rich lorest
                                          float: right;
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            Phasilic ingestion in the others the storm of the sear

guide site which code requirem it you accumpt did not

blacked and of others are traggle-god, or the sear

classe. Some lattic establishes agreed a skippe. Manis-

sels had, the lattic did common blacked in the story bearest

some above the search of the story of the search

some above the search of the search of the search

search of the search of the search of the search

formation above the search of the search of the search of

the search of the search of the search of the search of the

formation above the search of the search of the search of the

search of the search of the search of the search of the search of the

commonth.
        </style>
        </head>
  <body>
<h2>Float Right</h2>
```

In this example, the image will float to the right

 Lorem ipsum.

Property	Description
box-sizing	Defines how the width and height of an element are calculated: should they include padding and borders, or not
<u>clear</u>	Specifies what should happen with the element that is next to a floating element
float	Specifies whether an element should float to the left, right, or not at all
<u>overflow</u>	Specifies what happens if content overflows an element's box
overflow-x	Specifies what to do with the left/right edges of the content if it overflows the element's content area
<u>overflow-y</u>	Specifies what to do with the top/bottom edges of the content if it overflows the element's content area

CSS Layout - Horizontal & Vertical Align

```
<style>
                                .center {
  .center {
                                  display: flex;
                                                               .center p {
    margin: auto;
                                  justify-content: center;
                                                                line-height: 1.5;
    width: 60%;
    border: 3px solid #73AD21;
                                 align-items: center;
                                                                 display: inline-block;
    padding: 10px;
                                  height: 200px;
                                                                 vertical-align: middle;
    text-align: center;
                                  border: 3px solid green;
  </style>
                                    Paragraph 1 in the div.
  descemdamt combinator
                                                               Paragraph 1 in the div.
  <style>
                                    Paragraph 2 in the div.
                                                               Paragraph 2 in the div.
  div p {
                                    Paragraph 3 in the div.
    background-color: yellow;
                                                               Paragraph 3 in the div (insid
  </style>
                                   Paragraph 4. Not in a div.
                                                               Paragraph 4 in the div.
                                   Paragraph 5. Not in a div.
   Paragraph 1 in the div.
                                                               Paragraph 5. Not in a div.
   Paragraph 2 in the div.
   <section>Paragraph 3 in the div.</p:</pre>
                                                               Paragraph 6. Not in a div.
 Paragraph 4. Not in a div.
                                      Paragraph 1 in the div.
 Paragraph 5. Not in a div.
                                      Paragraph 2 in the div.
                                      <section>
                                        <!-- not Child but Descendant --
  child combinator >
                                        Paragraph 3 in the div (inside)
<style>
                                      Paragraph 4 in the div.
div > p {
 background-color: yellow;
                                    Paragraph 5. Not in a div.
</style>
```

Subsequent-sibling Combinator (~)

```
div ~ p {
  background-color: yellow;
```

The subsequent-sibling combinator selects all elements that are next siblings of a specified element.

Paragraph 6. Not in a div.

CSS Pseudo-classes

A pseudo-class is used to define a special state of an element. Style an element when a user moves the mouse over it Style visited and unvisited links differently Style an element when it gets focus Style valid/invalid/required/optional form elements

```
Pseudo-classes can be
selector:pseudo-class {
                             combined with HTML classes:
  property: value;
                             When you hover over the link
                             it will change color:
/* unvisited link */
                              a.highlight:hover {
a:link {
                                color: #ff0000;
  color: #FF0000;
                             An example of using the :hover
                             pseudo-class on a <div>
/* visited link */
                             element:
a:visited {
                             div:hover {
  color: #00FF00;
                               background-color: blue;
                            Hover over a <div> element to
                             show a  element (tooltip):
/* mouse over link */
                             p {
a:hover {
                               display: none;
  color: #FF00FF;
                               background-color: yellow;
                               padding: 20px;
```

div:hover p {

display: block;

Note: a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective! a:active MUST come after a:hover in the CSS definition in

Pseudo-class names are not case-sensitive.

/* selected link */

color: #0000FF;

a:active {

```
You can use the overflow
                           property when you want
                           to have better control of
background-color: coral;
                            the layout. The overflow
                           property specifies what
                            happens if content
                            overflows an element's
                            box.
```

You can use the overflow property when you want to have better control of

You can use the overflow property when you want to have better control of the layout. The overflow property specifies what div {

You can use the overflow property when you want to have better control of the layout. The

overflow: auto;

The clearfix Hack

div {

div {

div {

width: 200px;

height: 65px;

overflow: visible;

overflow: hidden;

overflow: scroll;

If a floated element is taller than the containing element, it will "overflow" outside of its container. We can then add a clearfix hack to solve this problem:

Without Clearfix

With Clearfix

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum.

The clear property can have one of the following values: none - The element is not pushed below left or right floated elements. This is default

left - The element is pushed below left floated elements .clearfix::after { right - The element is pushed below right floated elements content: ""; both - The element is pushed below both left and right floated elements clear: both; inherit - The element inherits the clear value from its parent display: table;

The overflow: auto clearfix works well as long as you are able to keep } control of your margins and padding (else you might see scrollbars).

CSS Layout - display: inline-block

Compared to display: inline, the major difference is that display: inline-block allows to set a width and height on the

Also, with display: inline-block, the top and bottom margins/paddings are respected, but with display: inline they

Compared to display: block, the major difference is that display: inline-block does not add a line-break after the element, so the element can sit next to other elements.

The display Property

display: inline

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consecuat scelerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam venenatis gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

lisplay: inline-block

/* If the text has multiple lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consecuat celerisque elit sit amet consequat. Aliquam erat volutpat. Aliquam

gravida nist sit amet facilisis. Nullam cursus fermentum velit sed venenatis laoreet.

display: block

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum consequat scelerisque elit sit amet consequat. Aliquam erat volutpat.

venenatis

gravida nisl sit amet facilisis. Nullam cursus fermentum velit sed laoreet.

selector::pseudo-element {

font-variant: small-caps;

You can use the ::first-

line pseudo-element to add a

special effect to the first

property: value;

p::first-line {

color: #ff0000;

line of a text.

<style>

</style>

</head>

</body>

<body>

CSS Opacity / Transparency

The **opacity** property specifies the opacity/transparency of an element.

```
img:hover {
                                         div {
img {
                                           opacity: 0.3;
  opacity: 0.5;
                       opacity: 1.0;
```

CSS Pseudo-elements

A CSS pseudo-element is used to style specific parts of an element. Style the first letter or line, of an element Insert content before or after an element Style the markers of list items Style the viewbox behind a dialog box

The ::first-line pseudo-element is used to add a special style to the first line of a text. The following example formats the first line of the text in all elements:

```
p::first-line {
  color: #ff0000;
  font-variant: small-caps;
```

</html> YOU CAN USE THE ::FIRST-LINE PSEUDO-ELEMENT TO ADD A SPECIAL EFFECT mor

to the first line of a text. Some more text. And even more, and more, and mor

more, and more, and more, and more, and more, and more, and		
more, and more, and more.	background	
Property	Description	
<u>background</u>	Sets all the background properties in one declaration	
background-attachment	Sets whether a background image is fixed or scrolls with the rest page	
background-clip	Specifies the painting area of the background	
background-color	Sets the background color of an element	
background-image	Sets the background image for an element	
background-origin	Specifies where the background image(s) is/are positioned	

bac Sets the starting position of a background image background-position background-repeat Sets how a background image will be repeated Specifies the size of the background image(s) background-size