HTML5& CSS3

Presentation by Uplatz

Contact Us: https://training.uplatz.com/
Email: info@uplatz.com
Phone:+44 7836 212635

Uplatz

Table Of Contents:

Overflow

OverflowValue	Details
visible	Shows all overflowing content outside the element
scroll	Hides the overflowing content and adds a scroll bar
hidden	Hides the overflowing content, both scroll bars disappear and the page becomes fixed
auto	Same as scroll if content overflows, but doesn't add scroll bar if content fits
inherit	Inherit's the parent element's value for this property

overflow-wrap

overflow-wrap tells a browser that it can break a line of text inside a targeted element onto multiple lines in anotherwise unbreakable place

- Helpful in preventing an long string of text causing layout problems due to overflowing it's container.

CSS

```
div {
 width:100px;
 outline: 1px dashed #bbb;
}
#div1 {
 overflow-wrap:normal;
}
#div2 {
 overflow-wrap:break-word;
}
```

HTML

```
<div id="div1">
 <strong>#div1</strong>: Small words are displayed normally, but a long word like <span style="red;">supercalifragilisticexpialidocious</span> is too long so it will overflow past the
edge of the line-break
</div>
<div id="div2">
 <strong>#div2</strong>: Small words are displayed normally, but a long word like <span style="red;">supercalifragilisticexpialidocious</span> will be split at the line break and continue
on the next line.
</div>
```

overflow-wrap – Value **Details**

normal Lets a word overflow if it is longer than the line

break-word Will split a word into multiple lines, if necessary

inherit Inherits the parent element's value for this property

overflow-x and overflow-y:

- These two properties work in a similar fashion as the overflow property and accept the same values.
- The overflow-x parameter works only on the x or left-to-right axis.
- The overflow-y works on the y or top-to-bottom axis.

HTML:

```
<div id="div-x">
 If this div is too small to display its contents,
 the content to the left and right will be clipped.
</div>
<div id="div-y">
 If this div is too small to display its contents,
 the content to the top and bottom will be clipped.
</div>
```

CSS:

```
div {
 width: 200px;
 height: 200px;
}
```

```
#div-x {
 overflow-x: hidden;
}
#div-y {
 overflow-y: hidden;
}
```

overflow: scroll

HTML

```
<div>
 This div is too small to display its contents to display the effects of the overflow property.
</div>
```

CSS

```
div {
 width:100px;
```

```
height:100px;
 overflow:scroll;
}
```

- The content above is clipped in a 100px by 100px box, with scrolling available to view overflowing content.
- Most desktop browsers will display both horizontal and vertical scrollbars, whether or not any content is clipped.
- This can avoid problems with scrollbars appearing and disappearing in a dynamic environment.
- Printers may printoverflowing content.

overflow: visible:

HTML

```
<div>
 Even if this div is too small to display its contents, the content is not clipped.
</div>
```

CSS

```
div {
 width:50px;
 height:50px;
 overflow:visible;
}
```

Block Formatting Context Created with Overflow:

- Using the overflow property with a value different to visible will create a new block formatting context.
- This is useful for aligning a block element next to a floated element.

CSS

```
img {
 float:left;
 margin-right: 10px;
}
 div {
 overflow:hidden; /* creates block formatting context
*/
}
```

HTML

```
<img src="http://placehold.it/100x100">
<div>
 <p>Lorem ipsum dolor sit amet, cum no paulo mollis
pertinacia.</p>
```

Ad case omnis nam, mutat deseruisse persequeris eos ad, in tollit debitis sea.

Floats:

Float an Image Within Text

- The most basic use of a float is having text wrap around an image.
- The below code will produce two paragraphs

and an image, with the second paragraph flowing around the image.

- Notice that it is always content after the floated element that flows around the floated element.

HTML:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi.

Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla.

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet.

clear property:

- The clear property is directly related to floats. Property Values:
- none - Default. Allows floating elements on both sides
- left - No floating elements allowed on the left side
- right - No floating elements allowed on the right side
- both - No floating elements allowed on either the left or the right side
- initial - Sets this property to its default value. Read about initial
- inherit - Inherits this property from its parent element. Read about inherit

```
<html>
<head>
<style>
img {
 float: left;
}
p.clear {
 clear: both;
}
</style>
</head>
<body>
<img
src="https://static.pexels.com/photos/69372/pexels-
photo-69372-medium.jpeg" width="100">
```

Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem

ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum

Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum

Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum

</body>

</html>

Clearfix

- clearfix is a concept (that is also related to floats, thus the possible confusion).

- To contain floats, you've to add .cf or .clearfix class on the container (the parent) and style this class

with a few rules described below.

In-line DIV using float

- The div is a block-level element, i.e it occupies the whole of the page width and the siblings are place one below the other irrespective of their width.

```
<div>
 <p>This is DIV 1</p>
</div>
<div>
 <p>This is DIV 2</p>
</div>
```

- We can make them in-line by adding a float css property to the div.

HTML:

```
<div class="outer-div">
 <div class="inner-div1">
 <p>This is DIV 1</p>
 </div>
 <div class="inner-div2">
 <p>This is DIV 2</p>
 </div>
</div>
```

CSS

```
.inner-div1 {
 width: 50%;
 margin-right:0px;
```

```
float:left;
 background : #337ab7;
 padding:50px 0px;
}
.inner-div2 {
 width: 50%;
 margin-right:0px;
 float:left;
 background : #dd2c00;
 padding:50px 0px;
}
p {
 text-align:center;
}
```

Use of overflow property to clear floats:

- Setting overflow value to hidden,auto or scroll to an element, will clear all the floats within that element.
- **Note**: using overflow:scroll will always show the scrollbox

Simple Two Fixed-Width Column Layout:

- A simple two-column layout consists of two fixed-width, floated elements.
- Note that the sidebar and content area

are not the same height in this example.

- This is one of the tricky parts with multi-column layouts using floats, and requires workarounds to make multiple columns appear to be the same height.

HTML:

```
<div class="wrapper">
<div class="sidebar">
 <h2>Sidebar</h2>
 <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio.</p>
</div>
<div class="content">
 <h1>Content</h1>
 <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.
Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque
nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin
ut ligula vel nunc egestas porttitor.
```

Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. </p>

CSS:

```
.wrapper {
 width:600px;
 padding:20px;
 background-color:pink;
 /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
 parent element to expand to contain its floated children. */
 overflow:hidden;
}
```

```
. sidebar {
 width:150px;
 float:left;
 background-color:blue
}
.content {
 width:450px;
 float:right;
 background-color:yellow;
}
```

Simple Three Fixed-Width Column Layout:

HTML:

```
<div class="wrapper">
 <div class="left-sidebar">
 <h1>Left Sidebar</h1>
```

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. </p>
 </div>
 <div class="content">
 <h1>Content</h1>
 <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.
Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque
nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin
ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non,
massa. </p>
 </div>
 <div class="right-sidebar">
```

```
<h1>Right Sidebar</h1>
 <p>Fusce ac turpis quis ligula lacinia aliquet.</p>
 </div>
</div>
```

CSS:

```
.wrapper {
 width:600px;
 background-color:pink;
 padding:20px;
 /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
 parent element to expand to contain its floated children. */
 overflow:hidden;
}
```

```
.left-sidebar {
 width:150px;
 background-color:blue;
 float:left;
}
.content {
 width:300px;
 background-color:yellow;
 float:left;
}
.right-sidebar {
 width:150px;
 background-color:green;
 float:right;
}
```

Two-Column Lazy/Greedy Layout:

- This layout uses one floated column to create a two-column layout with no defined widths.
- In this example the left sidebar is "lazy," in that it only takes up as much space as it needs.
- Another way to say this is that the left sidebar is "shrink-wrapped."
- The right content column is "greedy," in that it takes up all the remaining space.

HTML:

```
<div class="sidebar">
<h1>Sidebar</h1>
<img src="http://lorempixel.com/150/200/" />
</div>
<div class="content">
```

<h1>Content</h1>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla.

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem

. Proin
ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non,
massa. Fusce ac turpis quis ligula lacinia aliquet. Mauris ipsum. Nulla metus metus, ullamcorper
vel, tincidunt sed, euismod in, nibh. </p>

CSS:

```
.sidebar {
 /* `display:table;` shrink-wraps the column */
 display:table;
 float:left;
 background-color:blue;
}
```

```
.content {
 /* `overflow:hidden;` prevents `.content` from flowing under `.sidebar` */
 overflow:hidden;
 background-color:yellow;
}
```

Font Size

HTML:

```
<div id="element-one">Hello I am some text.</div>
<div id="element-two">Hello I am some smaller text.</div>
```

CSS:

```
#element-one {
 font-size: 30px;
}
```

```
#element-two {
 font-size: 10px;
}
```

- The text inside #element-one will be 30px in size, while the text in #element-two will be 10px in size

Text Transform:

- The text-transform property allows you to change the capitalization of text.
- Valid values are: uppercase,
- capitalize, lowercase, initial, inherit, and none

CSS:

```
.example1 {
 text-transform: uppercase;
}
```

```
.example2 {
 text-transform: capitalize;
}
.example3 {
 text-transform: lowercase;
}
```

HTML

```
<p class="example1">
 all letters in uppercase <!-- "ALL LETTERS IN UPPERCASE"
-->
</p>
<p class="example2">
 all letters in capitalize <!-- "All Letters In Capitalize
(Sentence Case)" -->
</p>
```

```
<p class="example3">
 all letters in lowercase <!-- "all letters in lowercase" -->
</p>
```

Word Spacing

The word-spacing property specifies the spacing behavior between tags and words.

Possible values:

- a positive or negative length (using em px vh cm etc.) or percentage (using %)
- the keyword normal uses the font's default word spacing
- the keyword inherit takes the value from the parent element

CSS

```
.normal { word-spacing: normal; }
```

```
.narrow { word-spacing: -3px; }
.extensive { word-spacing: 10px; }
```

HTML

```
<p>
 <span class="normal">This is an example, showing the effect of "word-spacing".</span><br>
 <span class="narrow">This is an example, showing the effect of "word-spacing".</span><br>
 <span class="extensive">This is an example, showing the effect of "word-spacing".</span><br>
</p>
```

Flexible Box Layout (Flexbox):

The Flexible Box module, or just 'flexbox' for short, is a box model designed for user interfaces, and it allows users to align and distribute space among items in a container such that elements behave predictably

- the page layout must accommodate different, unknown screen sizes.
- A flex container expands items to fill available space
- and shrinks them to prevent overflow.

Dynamic Vertical and Horizontal Centering (alignitems, justify-content)

Simple Example (centering a single element)

HTML

```
<div class="aligner">
 <div class="aligner-item">…</div>
</div>
```

CSS

```
.aligner {
 display: flex;
 align-items: center;
```

```
 justify-content: center;
}
.aligner-item {
 max-width: 50%; /*for demo. Use actual width instead.*/
}
```

Reasoning

Property	Value	Description
align-items	center	This centers the elements along the axis other than the one specified by flex-direction, i.e., vertical centering for a horizontal flexbox and horizontal centering for a vertical flexbox.
justify-content	**center**	This centers the elements along the axis specified by flex-direction. I.e., for a horizontal (flex-direction: row) flexbox, this centers horizontally, and for a vertical flexbox (flex-direction: column) flexbox, this centers vertically)

Individual Property Examples

- All of the below styles are applied onto this simple layout:

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

where #container is the flex-box.

Example: justify-content: center on a horizontal flexbox

CSS:

```
div#container {
 display: flex;
```

```
flex-direction: row;
 justify-content: center;
}
```

Example: justify-content: center on a vertical flexbox

CSS:

```
div#container {
 display: flex;
 flex-direction: column;
 justify-content: center;
}
```

Example: align-content: center on a horizontal flexbox

CSS:

```
div#container {
 display: flex;
```

```
 flex-direction: row;
 align-items: center;
}
```

Example: align-content: center on a vertical flexbox

CSS:

```
div#container {
 display: flex;
 flex-direction: column;
 align-items: center;
}
```

Example: Combination for centering both on horizontal flexbox

```
div#container {
 display: flex;
```

```
 flex-direction: row;
 justify-content: center;
 align-items: center;
}
```

Example: Combination for centering both on vertical flexbox

```
div#container {
 display: flex;
 flex-direction: column;
 justify-content: center;
 align-items: center;
}
```

Optimally fit elements to their container

- One of the nicest features of flexbox is to allow optimally fitting containers to their parent elem

HTML:

```
<div class="flex-container">
 <div class="flex-item">1</div>
 <div class="flex-item">2</div>
 <div class="flex-item">3</div>
 <div class="flex-item">4</div>
 <div class="flex-item">5</div>
</div>
```

CSS:

```
.flex-container {
 background-color: #000;
 height: 100%;
 display:flex;
 flex-direction: row;
```

```
flex-wrap: wrap;
 justify-content: flex-start;
 align-content: stretch;
 align-items: stretch;
}
.flex-item {
 background-color: #ccf;
 margin: 0.1em;
 flex-grow: 1;
 flex-shrink: 0;
flex-basis: 200px; /* or % could be used to ensure a specific layout */
}
```

Perfectly aligned buttons inside cards with Flexbox:

Uplatz

- It's a regular pattern in design these days to vertically align call to actions inside its containing cards like this:

HTML:

```
<div class="cards">
 <div class="card">
 <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
 <p><button>Action</button></p>
 </div>
 <div class="card">
 <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco
```

et ut sed qui minim.

Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco et ut sed qui minim.

Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco et ut sed qui minim.

Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco et ut sed qui minim.

Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim mollit labore dolore voluptate ullamco et ut sed qui minim.

<button>Action</button>

CSS

```
.cards {
 display: flex;
}.card {
 border: 1px solid #ccc;
 margin: 10px 10px;
 padding: 0 20px;
}
button {
 height: 40px;
 background: #fff;
 padding: 0 40px;
 border: 1px solid #000;
}p:last-child {
 text-align: center;
}
```

Summary:

- Overflow
- Floats
- Font Size
- Text Transform
- Flexible Box Layout (Flexbox)

Thank You……

If you have any quries please write to info@uplatz.com".