HTML Layout

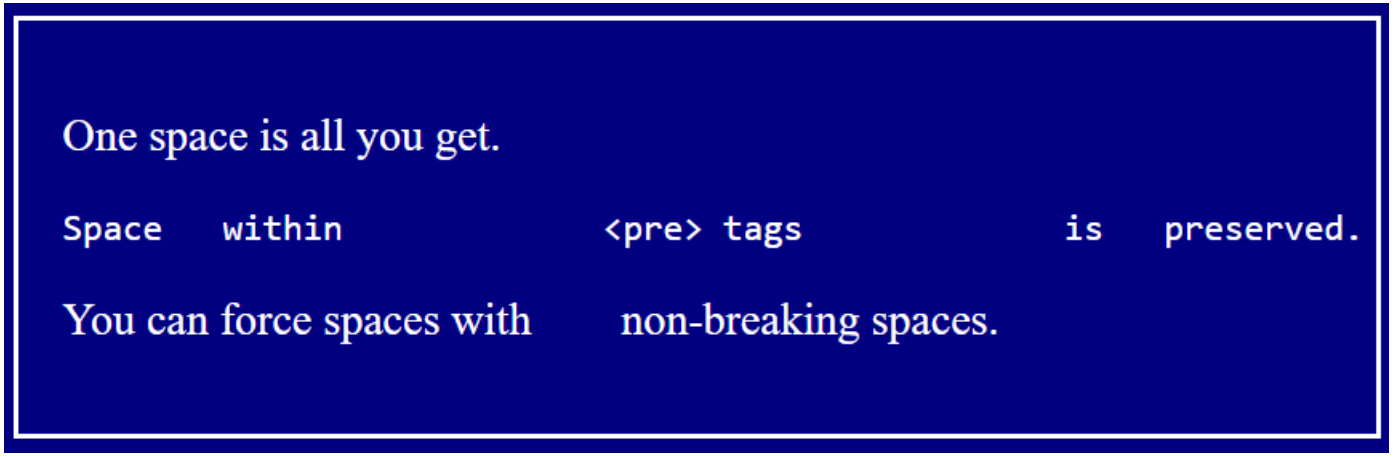
Topics Covered

- Whitespace and Entities
- Block Layout
- Inline Layout
- Table Layout
- Flexbox Layout
- Static Positioning
- Relative Positioning
- Absolute Positioning
- Fixed Positioning

HTML and Whitespace

Except in specific contexts, HTML ignores whitespace. This behavior can be customized using the white-space CSS property.

```
<div class="wrapper">
  <p>One space                    is                  all            you get.</p>
  <pre>Space   within              &lt;pre&gt; tags              is   preserved.</pre>
  <p>You can force spaces with        non-breaking spaces.</p>
</div>
```



HTML Entities

The ‘<’ and ‘>’ characters are reserved symbols in HTML, so to render them as literals we must represent them as entities:

```
&lt;pre&gt;
```

Entity notations begin with an ampersand and terminate with a semicolon. There are numerous HTML entities. The most common are:

Entity	Meaning
< and >	Less than ‘<’ and greater than ‘>’.
	Non-breaking space. The non-breaking space does not collapse like other whitespace, and it does not break a line.
"	A double-quote. Can be used within an attribute value.
&	& (ampersand)
© ®	© and ® symbols

Block-level Elements

Block elements occupy space, causing other elements to flow around them.

The prototypical block element is .

Empty, unstyled elements are invisible. But elements can be given css styles that cause them to occupy space:

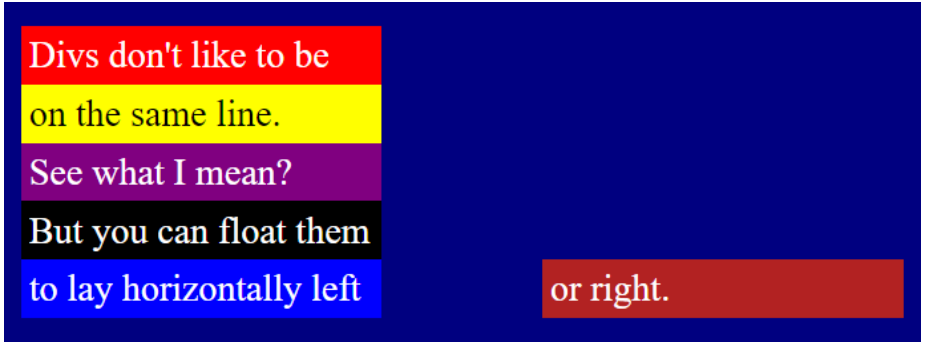
```
<div style="width:100px;height:30px;background-color:#f00"></div>
<div></div><div></div>  <!-- These are invisible -->
<p></p><p></p><p></p>   <!-- And so are these -->
```

This causes a 100 x 30 pixel red rectangle to render. The empty, unstyled elements are invisible.

Floating Blocks

Block elements like to occupy their own line, but css provides a float property that allows them to align left or right in the same horizontal space:

```
<div style="background-color:red">Divs don't like to be</div>
<div style="background-color:yellow;color:#000">on the same line.</div>
<div style="background-color:purple">See what I mean?</div>
<div style="background-color:black">But you can float them</div>
<div style="background-color:blue;float:left">to lay horizontally left</div>
<div style="background-color:firebrick;float:right">or right.</div>
```



Column Layouts

You can use floats to create multi-column pages.

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.

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. Quisque volutpat condimentum velit. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nam nec ante.

Sed lacinia, urna non tincidunt mattis, tortor neque adipiscing diam, a cursus ipsum ante quis turpis. Nulla facilisi. Ut fringilla. Suspendisse potenti. Nunc feugiat mi a tellus consequat imperdiet. Vestibulum sapien. Proin quam. Etiam ultrices. Suspendisse in justo eu magna luctus suscipit. Sed lectus. Integer euismod lacus luctus magna.

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.

CSS for Multi-Column Pages

```css
.leftcolumn {
  width: 15%;
  float: left;
  padding: 1em;
  font-size: 0.8em;
}
```

```css
.centercolumn {
  width: 60%;
  float: left;
  padding: 2em;
  border: solid 1px yellow;
}
```

```css
.rightcolumn {
  width: 15%;
  float:right;
  padding 1em;
  font-size: 0.8em;
}
```

```html
<div class="page">
  <h2>You can use floats to create multi-column pages.</h2>
  <div class="leftcolumn">...
  </div>
  <div class="rightcolumn">...
  </div>
  <div class="centercolumn">...
  </div>
  <!--After floating, we must clear the float behavior -->
  <div style="clear:both"></div>
</div>
<hr/>
```

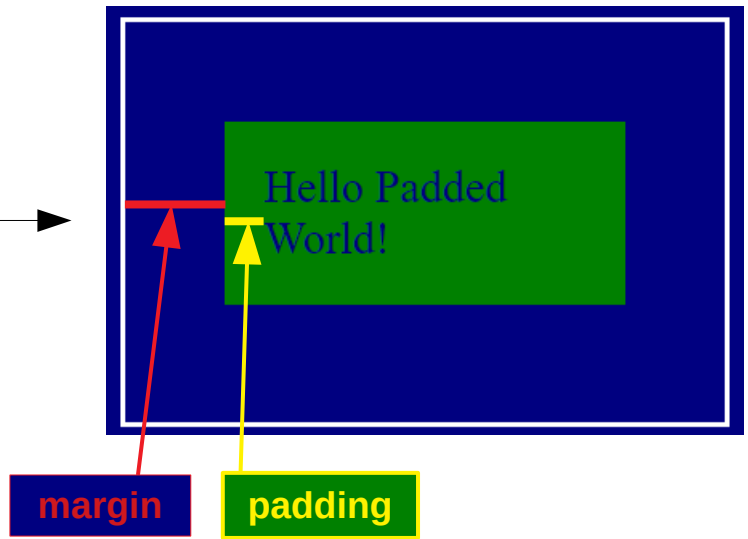
Margins and Padding

Block elements can be given margins to provide space around them.

Block elements can use padding to provide space around their content.

```
.ctr {
  width:300px;
  height: 200px;
  border: solid 2px white;
}
.content {
  background-color: green;
  color: navy;
  margin:50px;
  padding:20px;
}
```

```
<div class="ctr">
  <div class="content">
    Hello Padded World!
  </div>
</div>
```



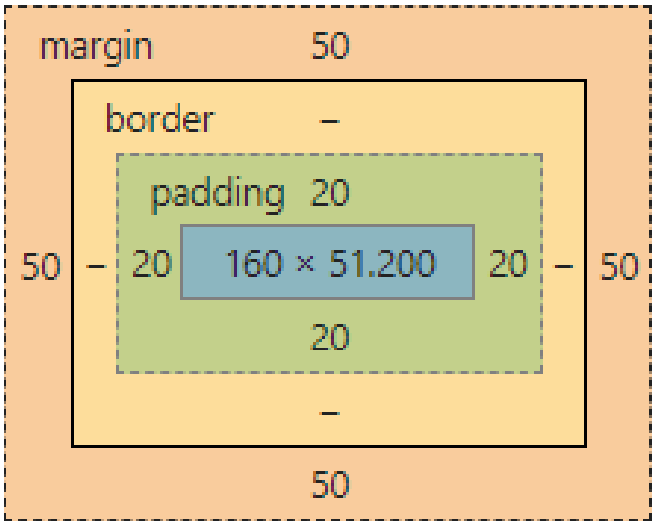
Margins/Padding: Browser Perspective

Our css content style defines a margin of 50px and padding of 20px.

When we select our content element in the Developer Tools, we see a graphical representation of the layout defined by our css.

```
.content {
  background-color: green;
  color: navy;
  margin:50px;
  padding:20px;
}
```

```
▼<div class="ctr">
    <div class="content">
                    Hello Padded World!
                </div> == $0
  </div>
```



Margin and Padding T R B L

For finer control over the layout, CSS allows us to define left, right, top and bottom margins/padding independently:

```css
.trbl {
  margin-top: 5px;
  margin-bottom: 2em;
  margin-left: 100px;
  margin-right: 20%;
}
.trbl2 {
  margin: 5px 2em 100px 20%;
}
```

Mnemonic: Think of a clock, and the values start at midnight.

The classes trbl and trbl2 are identical in their effect. Also note the different units that we can use for these layout properties.

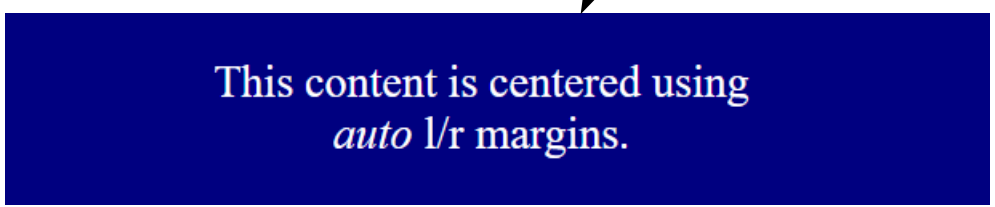
Margin Auto

A common requirement is to center an element horizontally within its container. For this, CSS defines the *auto* value:

```css
.ctr2 {
  width: 100%;
  margin: 5%;
  border solid 2px white;
}
.center {
  width:300px;
  margin: 0 auto;
  text-align: center;
}
```

+

```html
<div class="ctr2">
  <div class="center">
    This content is centered using <i>auto</i> l/r margins.
  </div>
</div>
```



Other Block Elements

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>		<dl>
<dt>	<fieldset>	<figcaption>	<figure>	<footer>	<form>	<h1>-<h6>	<header>
<hr>		<main>	<nav>	<noscript>		<output>	<p>
<pre>	<section>	<table>	<tfoot>		<video>		

Inline Layout

Inline elements do not start a new line, and they occupy only as much width as required.

The prototypical inline element is . Spans are typically used to format text:

```
<p>The &lt;span&gt; element is used to apply
  <span style="color:red;font-style:italic;">particular styling</span>
  to a stretch of text.
</p>
```

The element is used to apply *particular styling* to a stretch of text.

Flowing Text Around Blocks

Because inline content flows, it can wrap around a floated block:

```
nostra, per inceptos himenaeos. Nam nec ante. Sed lacinia, urna non tincidunt mattis, tortor neque adipiscing diam, a
<div class="hilite">
  This text acquires special emphasis because of its style &amp;amp; position and informs you of some important aspect of
  the surrounding content.
</div>
cursus ipsum ante quis turpis. Nulla facilisi. Ut fringilla. Suspendisse potenti. Nunc feugiat mi a tellus consequat
```

```
.hilite {
  float:right;
  margin: 1em 0  1em 1em;
  width: 35%;
  background-color: beige;
  color:navy;
  padding:1em;
  border: solid 2px red;
}
```

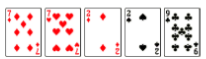
aliquet. Mauris ipsum. Nulla metus metus, ullamcorper vel, tincidunt sed,
litora torquent per conubia nostra, per inceptos himenaeos. Nam nec ante.
ante quis
consequat
agna luctus
ae pharetra
sum primis
i. Praesent
ipsum sit

This text acquires special emphasis because of its style & position and informs you of some important aspect of the surrounding content.

augue. Vestibulum tincidunt malesuada tellus. Ut ultrices ultrices enim.
. Integer lacinia sollicitudin massa. Cras metus. Sed aliquet risus a tortor.

HTML Tables

HTML tables provide a structure for presenting tabular information, where our understanding of that information can be enhanced based on its location in a specific row and column.

Rank	Name	Example	Description
1	High Card		In a *High Card* hand, none of the stronger hands can be found. The hand is ranked based on the highest card. The highest-ranked card in this particular hand is King.
2	Pair		Exactly one of the ranks in this hand occurs twice, and none of the cards forms a straight or a flush. The hand is ranked based on the rank of the pair. A tie is broken based on the ranks of the remaining cards. This particular hand contains two cards of rank 10.
3	Two Pair		Exactly two of the ranks in this hand occurs twice, and none of the cards forms a straight or a flush. The hand is ranked based on the ranks of the top pair, then the second pair, and in case of a tie, the rank of the remaining card.
4	Three of a Kind		One of the ranks in this hand occurs exactly three times, and none of the cards forms a straight or a flush. The hand is ranked based on the rank of the triplet. Ties are broken by the ranks of the remaining cards.
5	Straight		Five or more cards in this hand are incrementally ranked. The strength of the straight is judged based on the highest card of the straight. The hand shown here is an 8-high straight.
6	Flush		Five or more cards of this hand are of the same suit, and none of the same-suited cards forms a straight. The strength of the hand is judged by the rank of the highest suited card. The hand shown here is a jack-high flush.
7	Full House		This hand consists of three cards of one rank and two cards of another. The strength of the hand is judged first by the tripled rank and then, if that ties, the paired rank.
8	Four of a Kind		This hand includes all 4 cards of a rank and one additional card. The strength of the hand is based on the rank of the quad. Ties are broken by the rank of the remaining card.
9	Straight Flush		This hand combines the qualities of straights and flushes: five or more cards are incrementally ranked, and each of these cards is of the same suit. The strength of the hand is determined by the highest card.
10	Royal Flush		This hand is technically a straight flush but is given its own category because it is the strongest possible poker hand.

	Suited												
	A	K	Q	J	10	9	8	7	6	5	4	3	2
A	+0.355	+0.179	+0.171	+0.164	+0.157	+0.141	+0.134	+0.125	+0.116	+0.117	+0.108	+0.099	+0.091
K	+0.162	+0.327	+0.144	+0.137	+0.130	+0.114	+0.098	+0.092	+0.084	+0.077	+0.068	+0.059	+0.050
Q	+0.154	+0.125	+0.302	+0.115	+0.108	+0.091	+0.076	+0.061	+0.055	+0.047	+0.038	+0.030	+0.021
J	+0.146	+0.117	+0.094	+0.278	+0.089	+0.072	+0.057	+0.042	+0.026	+0.021	+0.012	+0.003	-0.006
10	+0.139	+0.110	+0.087	+0.067	+0.253	+0.057	+0.042	+0.027	+0.011	-0.005	-0.012	-0.020	-0.029
9	+0.121	+0.092	+0.069	+0.049	+0.033	+0.224	+0.028	+0.013	-0.003	-0.018	-0.037	-0.043	-0.052
8	+0.113	+0.076	+0.053	+0.033	+0.017	+0.002	+0.195	+0.002	-0.013	-0.029	-0.047	-0.065	-0.072
7	+0.105	+0.069	+0.036	+0.016	-0.000	-0.014	-0.025	+0.165	-0.020	-0.036	-0.054	-0.072	-0.091
6	+0.094	+0.061	+0.030	-0.000	-0.017	-0.031	-0.042	-0.050	+0.135	-0.040	-0.058	-0.076	-0.094
5	+0.096	+0.053	+0.022	-0.006	-0.034	-0.048	-0.059	-0.066	-0.070	+0.105	-0.056	-0.073	-0.092
4	+0.086	+0.043	+0.012	-0.015	-0.041	-0.067	-0.078	-0.085	-0.089	-0.087	+0.071	-0.084	-0.102
3	+0.076	+0.034	+0.003	-0.025	-0.050	-0.074	-0.098	-0.105	-0.109	-0.106	-0.117	+0.037	-0.111
2	+0.067	+0.024	-0.007	-0.035	-0.060	-0.084	-0.104	-0.125	-0.129	-0.126	-0.136	-0.145	+0.002
	Non-Suited												

Element	Usage	Description
<table>	required	Container for tabular content.
<caption>	optional	Provides a title for the table. It may contain simple text or any other flow content.
<colgroup>	optional	Defines groups of columns for css formatting.
<col>	optional	Defines styling for a column
<thead>	optional	Defines a set of rows comprising the table header.
<tr>	required	Defines a row of cells in a table.
<th>	optional	Defines a header table cell.
<tbody>	optional	Defines a set of rows comprising the body of the table.
<td>	required	Defines a table cell. It may contain simple text or any other flow content.
<tfoot>	optional	Defines a set of rows summarizing the table content.

A Simple HTML Table

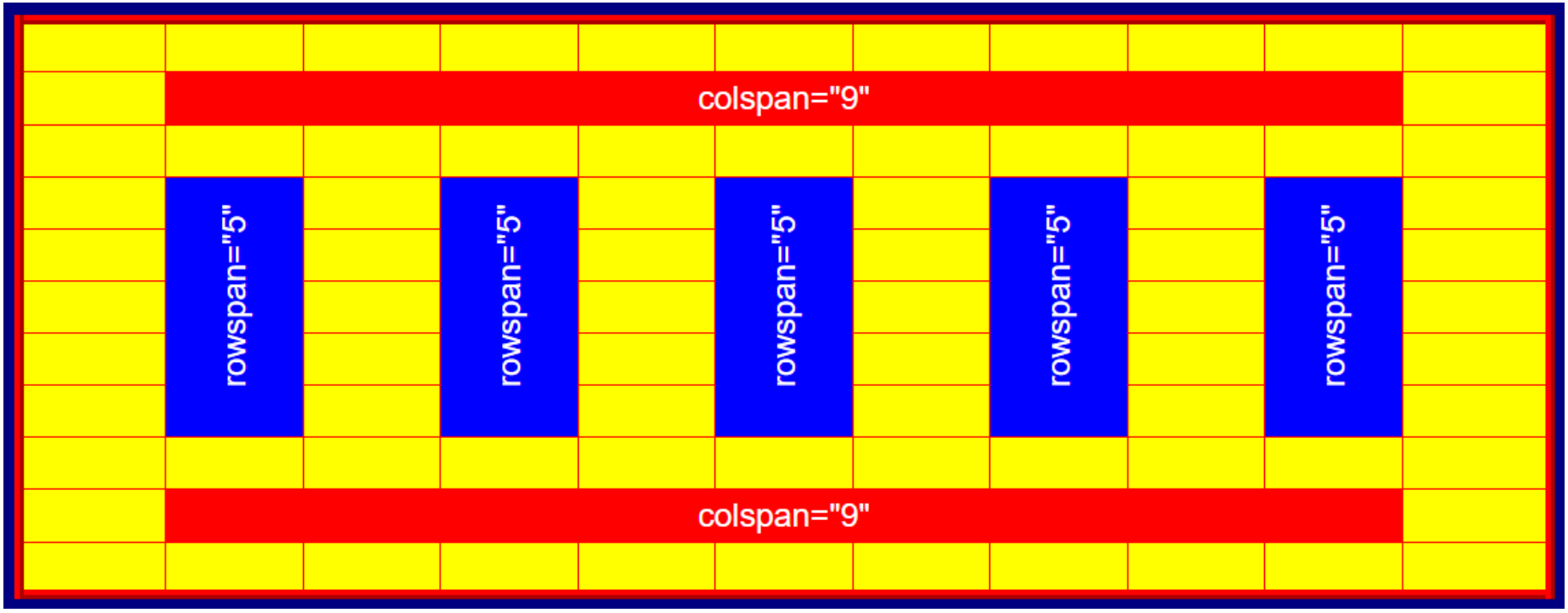
```
<table style="width:40%">
  <caption>A Simple HTML Table</caption>
  <tr>
    <th>Row #</th><th>Column A</th><th>Column B</th><th>Column C</th>
  </tr>
  <tr>
    <td>Row 1</td><td>Cell 1A</td><td>Cell 1B</td><td>Cell 1C</td>
  </tr>
  <tr>
    <td>Row 2</td><td>Cell 2A</td><td>Cell 2B</td><td>Cell 2C</td>
  </tr>
  <tr>
    <td>Row 3</td><td>Cell 3A</td><td>Cell 3B</td><td>Cell 3C</td>
  </tr>
</table>
```

A Simple HTML Table

Row #	Column A	Column B	Column C
Row 1	Cell 1A	Cell 1B	Cell 1C
Row 2	Cell 2A	Cell 2B	Cell 2C
Row 3	Cell 3A	Cell 3B	Cell 3C

Merging Cells

Table cells can be merged using the colspan and rowspan attributes on the <td> or <th> elements.



Using colspan

This row has 11 cells

This row has 3 cells

This row has 11 cells

```
<caption>Merging Cells</caption>
<tr>
  <td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/><td style="background-color: red" colspan="9">
    colspan="9"
  </td><td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/>
```

Using rowspan

```
<tr>
  <td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/>
  <td rowspan="5" style="background-color: blue">
    <span class="rot">rowspan="5"</span>
  </td>
  <td/>
  <td rowspan="5" style="background-color: blue">
    <span class="rot">rowspan="5"</span>
  </td>
  <td/>
  <td rowspan="5" style="background-color: blue">
    <span class="rot">rowspan="5"</span>
  </td>
  <td/>
  <td rowspan="5" style="background-color: blue">
    <span class="rot">rowspan="5"</span>
  </td>
  <td/>
  <td rowspan="5" style="background-color: blue">
    <span class="rot">rowspan="5"</span>
  </td>
  <td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/>
```

For each cell defining rowspans of 5, cells must be removed from the next 4 rows.

```
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
  <td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/><td/>
</tr>
<tr>
```

Use Tables Only for Tabular Information

Statistical Distributions

Distribution	Distribution Function	Density Function	Mean	Median
Arcsine	$F(x) = \frac{2}{\pi}\arcsin(\sqrt{x}) = \frac{\arcsin(2x-1)}{\pi} + \frac{1}{2}$	$f(x) = \frac{1}{\pi\sqrt{x(1-x)}}$	½	1&fasl;2
Cauchy	$\frac{1}{\pi\gamma\left[1+\left(\frac{x-x_0}{\gamma}\right)^2\right]}$	$\frac{1}{\pi}\arctan\left(\frac{x-x_0}{\gamma}\right)+\frac{1}{2}$	undefined	x_0
Gaussian	$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\frac{1}{2}\left[1+\operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right]$	μ	μ
Logit-normal	$\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(\operatorname{logit}(x)-\mu)^2}{2\sigma^2}}\frac{1}{x(1-x)}$	$\frac{1}{2}\left[1+\operatorname{erf}\left(\frac{\operatorname{logit}(x)-\mu}{\sqrt{2\sigma^2}}\right)\right]$	no analytical solution	$P(\mu)$
Poisson	$\frac{\lambda^k e^{-\lambda}}{k!}$	$\frac{\Gamma(\lfloor k+1\rfloor,\lambda)}{\lfloor k\rfloor!}$	λ	$\approx \lfloor \lambda + 1/3 - 0.02/\lambda \rfloor$
Rayleigh	$\frac{x}{\sigma^2}e^{-x^2/(2\sigma^2)}$	$1-e^{-x^2/(2\sigma^2)}$	$\sigma\sqrt{\frac{\pi}{2}}$	$\sigma\sqrt{2\ln(2)}$
Uniform Continuous	$\begin{cases}\frac{1}{b-a} & \text{for } x\in[a,b]\\ 0 & \text{otherwise}\end{cases}$	$\begin{cases}0 & \text{for } x<a\\ \frac{x-a}{b-a} & \text{for } x\in[a,b)\\ 1 & \text{for } x\geq b\end{cases}$	$\frac{1}{2}(a+b)$	$\frac{1}{2}(a+b)$

Many web developers have been tempted into using tables for page layouts. This is partly due to weaknesses in earlier versions of CSS.

Don't do it! Instead, CSS and the other techniques described in this presentation.

Flexbox Layouts

One can do a lot using floating divs and table layouts, but a container-based approach to layouts was (until recently) missing from HTML. The idea is to allow a container element to adjust the positioning of its child elements in a flexible manner. Thus the flexbox.

Let's begin with the CSS needed to create a flexbox:

```
.container {
  display: flex;
  border: solid 2px red;
  padding: 5px;
}
```

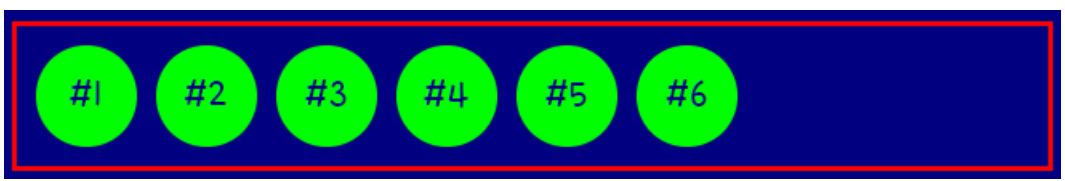
The CSS display style is powerful in many ways. Here we use it to enable the flex behavior.

CSS: display: flex

Now let's create a container and put a few child elements inside that container.

```
<div class="container">
  <div class="orb"><span>#1</span></div>
  <div class="orb"><span>#2</span></div>
  <div class="orb"><span>#3</span></div>
  <div class="orb"><span>#4</span></div>
  <div class="orb"><span>#5</span></div>
  <div class="orb"><span>#6</span></div>
</div>
```

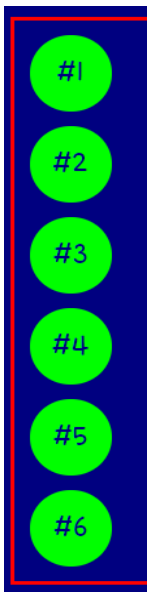
We've seen previously that divs like to occupy their own line, but the flex container causes these divs line up as if they were all floated left:

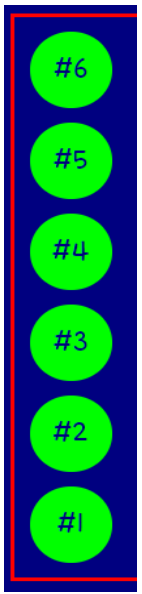


CSS: flex-direction

This behavior is determined by the flex-direction CSS style. The style defaults to ‘row’, but we can change it to ‘column’ to stack the divs vertically, ‘row-reverse’ to reverse the row, or ‘column-reverse’ to stack them inverted:

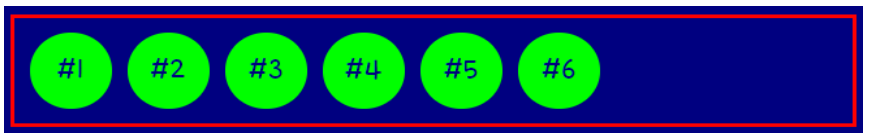
column

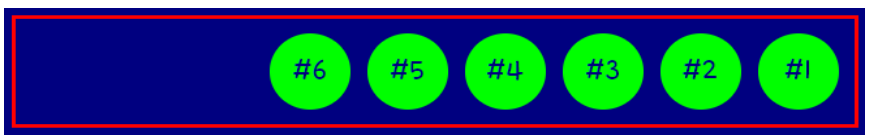




column-reverse

row

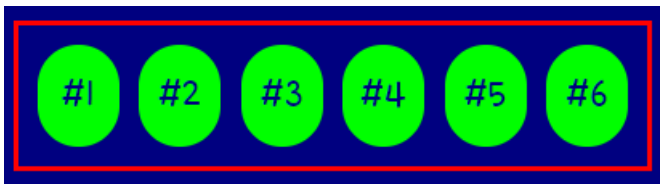




row-reverse

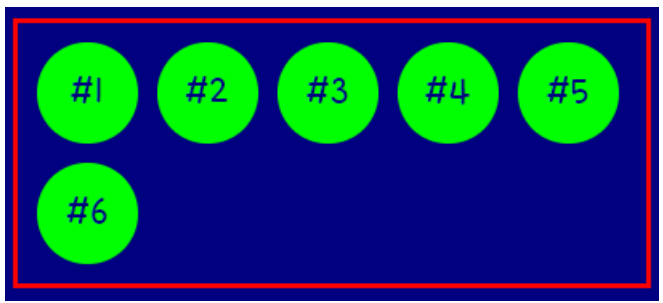
CSS: flex-wrap

By default, our divs will not wrap. Shrinking the browser to a minimum width just compresses them:

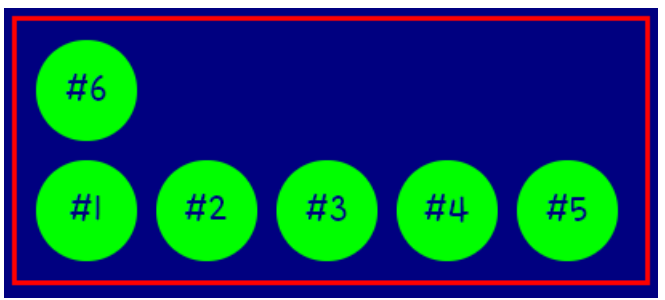


The default flex-wrap style is ‘nowrap’, but we can also use either ‘wrap’ or ‘wrap-reverse’:

```
flex-wrap: wrap;
```



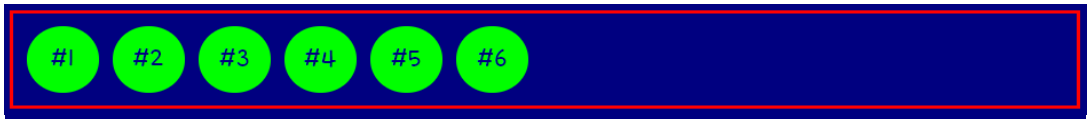

```
flex-wrap: wrap-reverse;
```



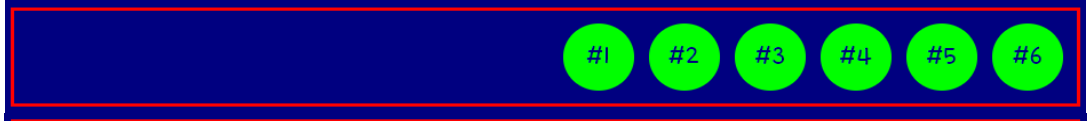
CSS: justify-content

We can specify the flex direction and wrapping. We can also specify how we want child elements to fill the available space using the justify-content CSS property:

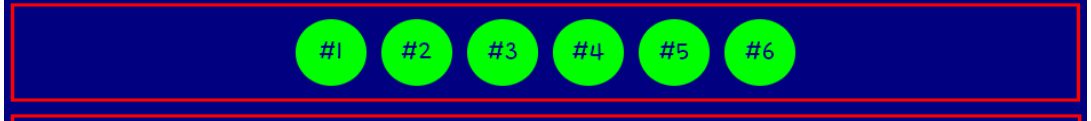
```
justify-content: flex-start;
```




```
justify-content: flex-end;
```

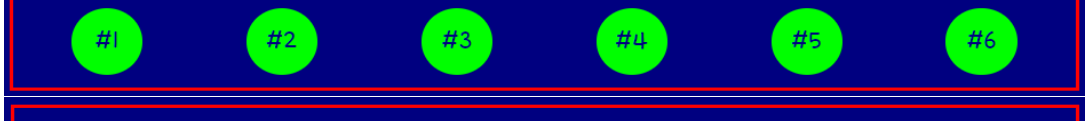



```
justify-content: center;
```

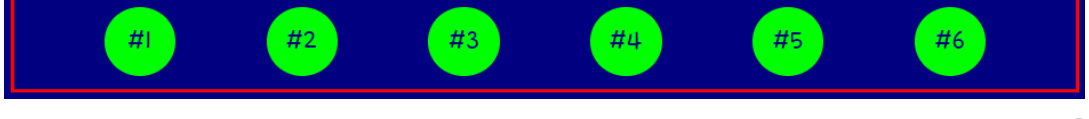



```
justify-content: space-between;
```


```
justify-content: space-around;
```




```
justify-content: space-evenly;
```



Additional Flex Styles

There are additional flex styles beyond the scope of this presentation. These are summarized here:

CSS Style	Applies to:	Description
align-items	container	Defines how items are laid out along the cross-axis (the axis perpendicular to the one specified by flex-direction).
align-content	container	Defines how the browser distributes space between and around content items along the cross-axis of their container. This only affects items that have wrapped.
order	child	Determines the order in which child items appear in the flex container.
flex-grow	child	Defines the ability for a flex item to grow to fill available space. It accepts a unitless value that serves as a proportion relative to the other items.
flex-shrink	child	Defines the ability for a flex item to shrink if necessary
flex-basis	child	Defines the default size of an element before the remaining space is distributed.
align-self	child	Overrides an items alignment as specified by the container's *align-items* property.

CSS Positioning

CSS defines a *position* style which lets us position elements either relative to the normal flow of the document or outside of the normal document flow.

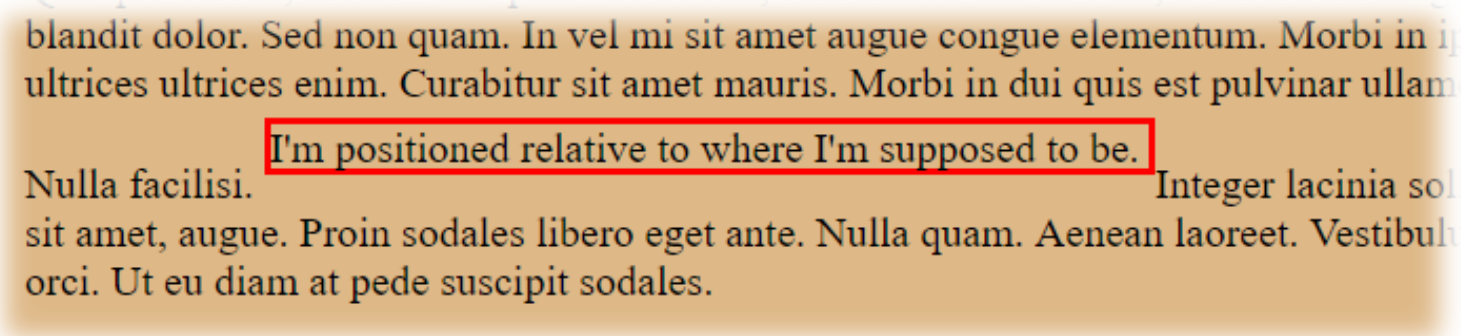
This style works in conjunction with the top, right, bottom and left styles.

Position value	Effects
static	The element is positioned according to the normal flow of the document. This is the default.
relative	The element is positioned in the normal flow of the document and is then offset by amounts based on top, right, bottom and left.
absolute	The element is taken out of the normal document flow and is positioned relative to its nearest positioned ancestor by amounts based on top, right, bottom and left.
fixed	The element is taken out of the normal document flow and displayed in the viewport based on top, right, bottom, and left.

Relative Positioning

```
.rel {
  position:relative;
  top: -14px;
  border: solid 2px red;
}
```

```
<p>Nulla facilisi.
<span class="rel">
  I'm positioned relative to where I'm supposed to be.
</span>
Integer lacinia sollicitudin massa. Cras metus. Sed alic
```



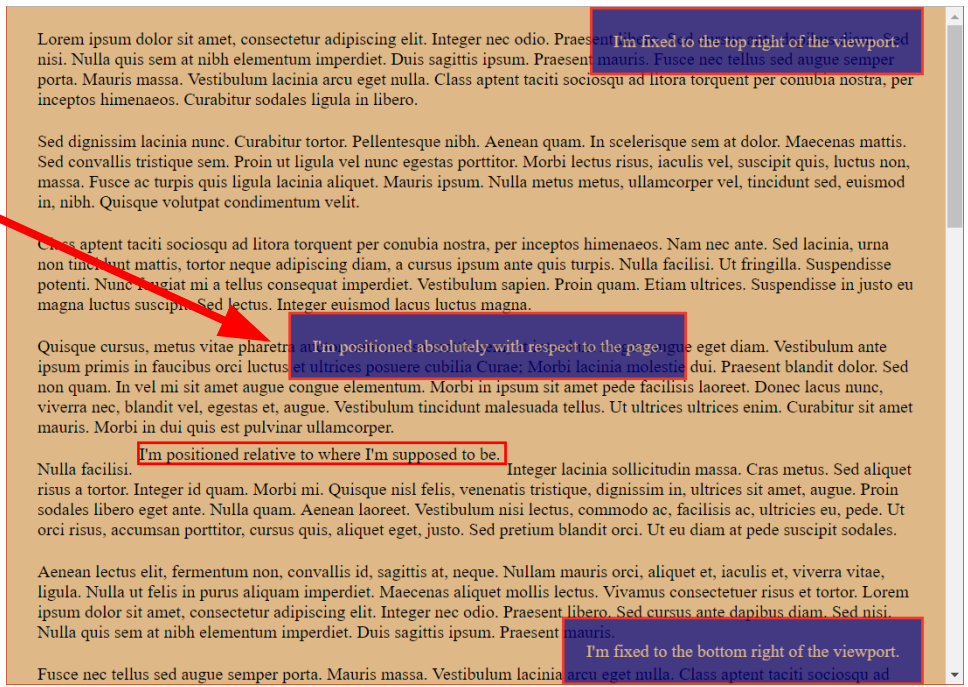
Absolute Positioning

```
.abs {
  position:absolute;
  left:30%;
  top:45%;
}
```

```
<div class="abs inset">
  I'm positioned absolutely with respect to the page.
</div>
```

If no positioned ancestor is found, the element is positioned relative to the <html> element.

This element will scroll with the page.

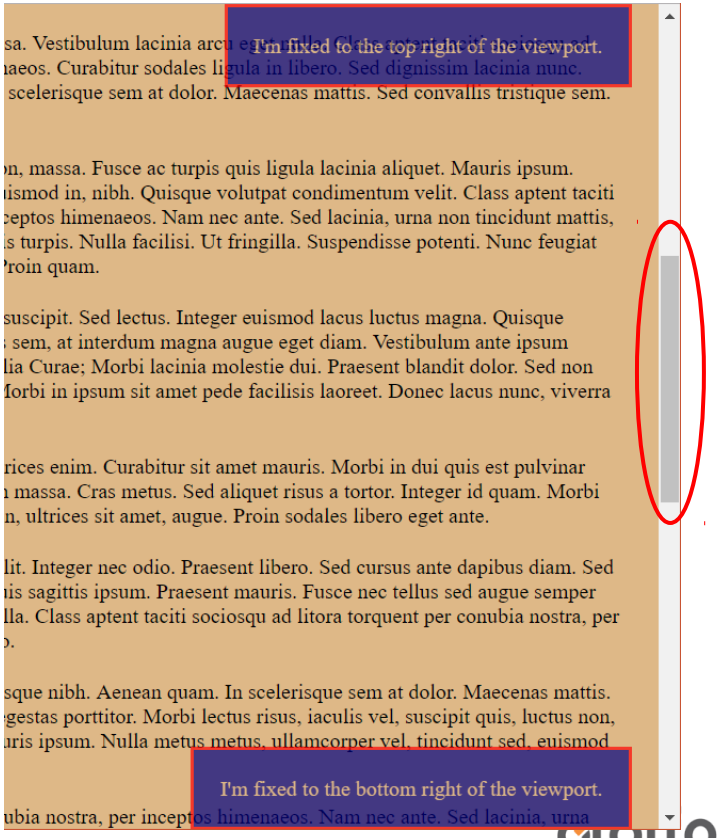


Fixed Positioning

```css
.fixtop {
  position: fixed;
  top: 0;
  right:20px;
}
.fixbot {
  position: fixed;
  bottom: 0;
  right: 20px;
}
```

No matter the position of the scroll-bar, these elements will be fixed to their locations within the viewport.

```html
<div class="fixtop inset">
  I'm fixed to the top right of the viewport.
</div>
<div class="fixbot inset">
  I'm fixed to the bottom right of the viewport.
</div>
```



Summary: HTML Layout

- Whitespace and Entities
- Block Layout
- Inline Layout
- Table Layout
- Flexbox Layout
- Static Positioning
- Relative Positioning
- Absolute Positioning
- Fixed Positioning