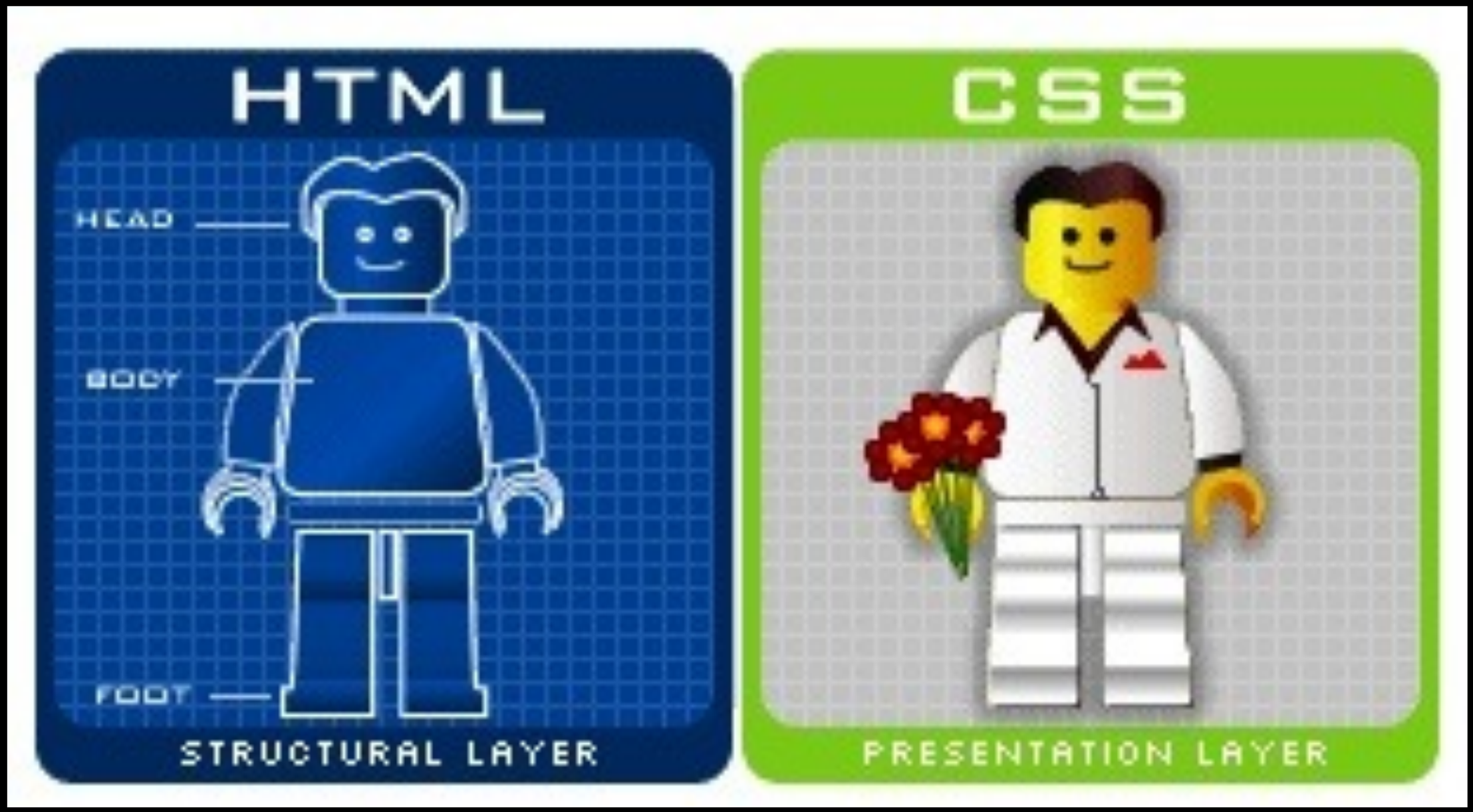
Cascading Style Sheets {CSS}

Prof. Jayakumar Sadhasivam Ph.D.
School of Computer Science and Engineering,
Vellore Institute of Technology, India.

Table of Content

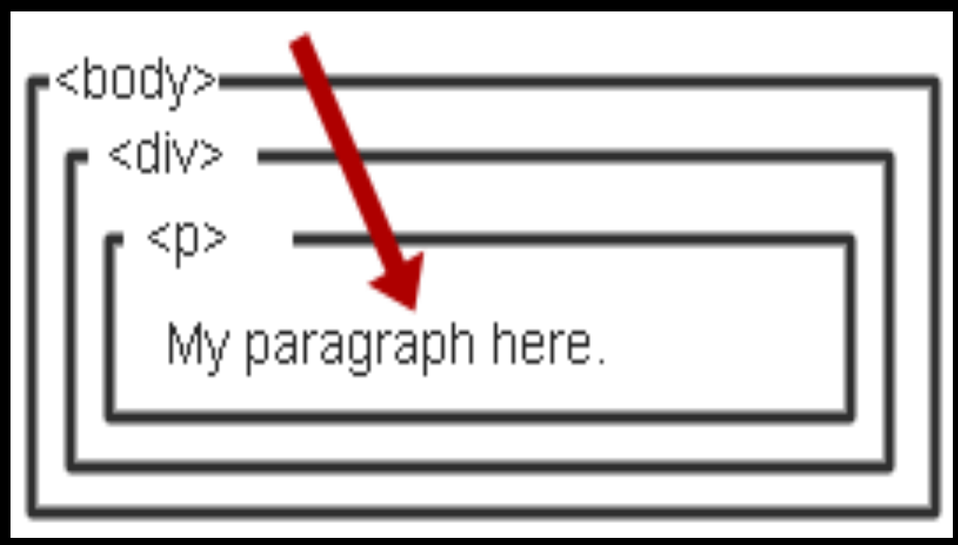
Prof. Jayakumar Sadhasivam, VIT, India.

your {CSS} to my <HTML>



CSS

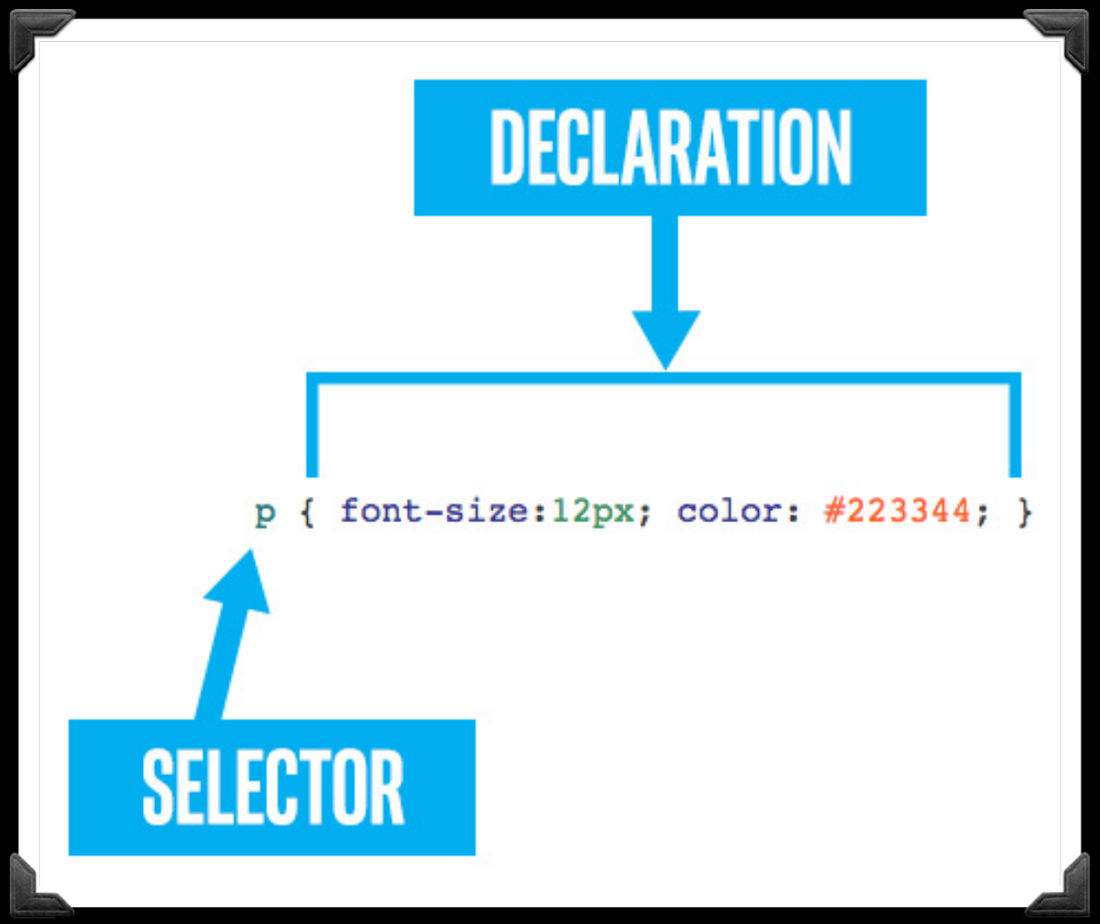
- Using CSS, we can determine the visual appearance of our HTML elements independent of the HTML itself.
- Standards-based set of properties and attributes to define styles
- To describe the presentation a document written in a ‘markup language’ like HTML or XML
  - ✓ Markup encoding: My paragraph here.
  - ✓ Defines the style of how things in tags appear.
  - ✓ Font, color, size, margins, etc.
- Cascading
  - ✓ Rules to determine how to apply markup that contains other markup



CSS Syntax

Selector

The selector is p. When a selector appears unprefixed by any punctuation, then it is assumed to match to an HTML tag. Thus, the p selector will apply the CSS rule to all <p> tags in the document.



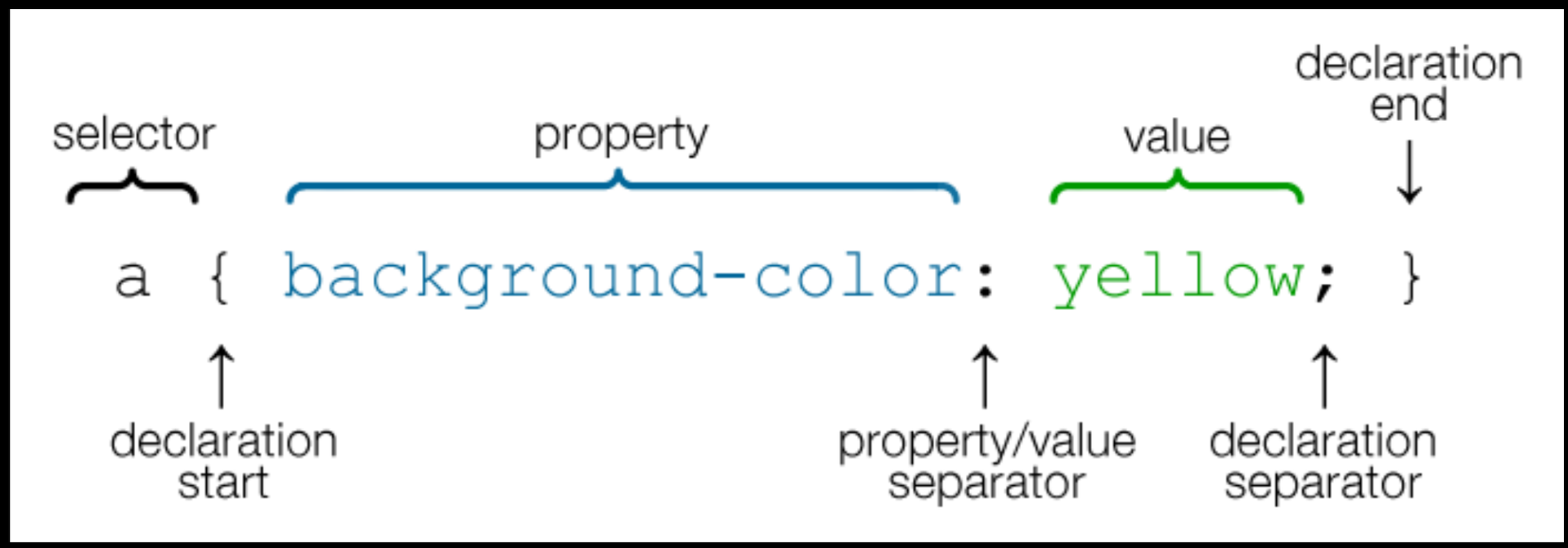
Declaration:

The declaration part of a CSS rule opens and closes with curly braces: { } And between them, go any number of property value pairs.

CSS Syntax

3 Elements to a CSS Statement

- Selector
  - ✓ What HTML sections does it affect?
- Property
  - ✓ What attribute of that HTML section will be affected?
- Value
  - ✓ What change will be made to that attribute?



CSS

Three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

- With an external style sheet, you can change the look of an entire Web site by changing one file.
- Each page must link to the style sheet using the **<link> tag**.
- The <link> tag goes inside the head section:

```
The <link> tag goes inside the head section:
<head>
  <link rel="stylesheet" type="text/css"
      href="mystyle.css" />
</head>
```

External Style Sheet

- The file should not contain any **HTML tags**.
- Style sheet should be saved with a **.css** extension.
- Example:

```
hr {color:red;}
p {margin-left:20px;}
body {background-image:url("bg.gif");}
```

Internal Style Sheet

- An internal style sheet should be used when a single document has a unique style.
- Define internal styles in the head section of an HTML page, by using the <style> tag
- Example:

```
<head>
<style type="text/css">
   hr {color:red;}
   p {margin-left:20px;}
   body {background-image:url(“back40.jpg");}
</style>
</head>
```

Inline Styles

- An inline style loses many of the advantages of style sheets by mixing content with presentation.
- To use inline styles you use the style attribute in the relevant tag.
- The style attribute can contain any CSS property.
- The example shows how to change the color and the left margin of a paragraph

```
<p style="color:red; margin-left:20px">This
    is a paragraph.</p>
```

CSS Comments

- CSS can include "comments" as well, by which you, the developer today, can leave notes and reminders to you, a different developer tomorrow. Or to others who might read your CSS.
- Comments begin with `/*` and **must** end with `*/` and they can span several lines. But they **cannot** be nested.

```
p {
   font-size: 8px;    /* small text. */
  }
```

CSS – Measurements

CSS – Measurements

- CSS supports a number of measurements including absolute units such as inches, centimetres, points, and so on, as well as relative measures such as percentages and em units. These values are needed while specifying various measurements in the style rules like **border = "1px solid red"**.

Unit	Description	Example
%	Defines a measurement as a percentage relative to another value, typically an enclosing element.	p {font-size: 16pt; line-height: 125%;}
cm	Defines a measurement in centimeters.	div {margin-bottom: 2cm;}
in	Defines a measurement in inches.	p {word-spacing: .15in;}
mm	Defines a measurement in millimeters.	p {word-spacing: 15mm;}
pt	Defines a measurement in points. A point is defined as 1/72nd of an inch.	body {font-size: 18pt;}
px	Defines a measurement in screen pixels.	p {padding: 25px;}

CSS SELECTOR

CSS SELECTOR

- The selector is used to select which elements in the HTML page will be given the styles inside the curly braces.
  - ✓ Types of Selectors
    - ✓Element
    - ✓Id
    - ✓Class
    - ✓position in document

Types of Selectors: **element**

```
<style>
    p{
      }

</style>
```

Select all **p** elements in the entire document.

Types of Selectors: **id**

```
<style>
    #header {
            }
</style>
```

- Selects any element with the id “header", e.g.
  `<p id=“header”> </p>`
- Element ids are unique, so that should only be one element.
- The "**#**" is how you tell CSS "this is an id."

Types of Selectors: **Class**

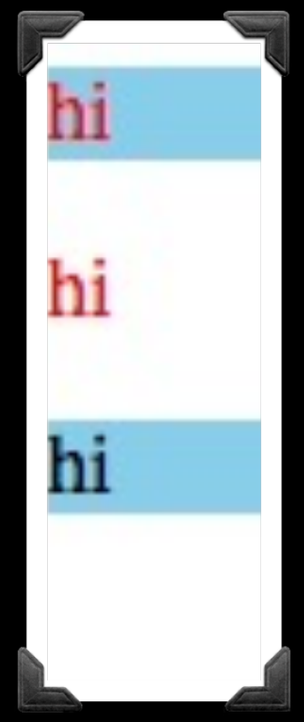
```
<style>
     .warning {
                    color: red;
              }
</style>
```

- Selects any element with the class name “warning”, e.g.
-
- Multiple elements can have the same class name.
- The "." is how you tell CSS "this is a class name."

Types of Selectors: **Class**

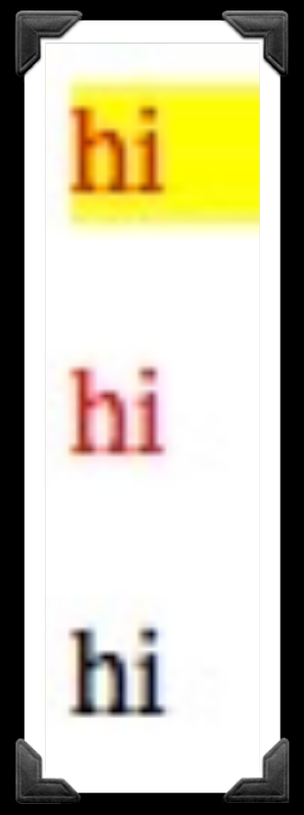
- An element can have only 1 id but multiple classes, so you can take advantage of that for greater flexibility.

```
<style>
.intro {
    background-color: skyblue;
}
.desc {
    color: red;
}
</style>
<p class="desc intro">hi</p>
<p class="desc">hi</p>
<p class=“intro">hi</p>
```



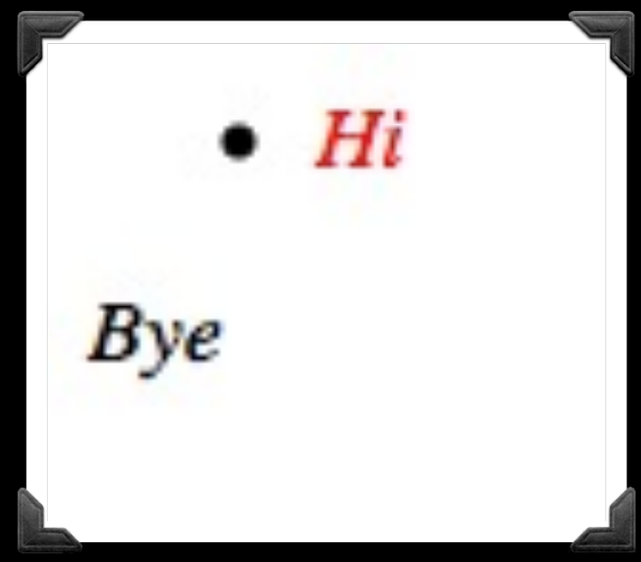
```
<style>
.asc{
    background-color: yellow;
}
.intro {
    background-color: blue;
}
.desc {
    color: red;
}
</style>
<p class="desc intro asc">hi</p>
<p class="desc">hi</p>
<p class=“intro">hi</p>
```

```
<style>
.intro {
    background-color: blue;
}
.desc {
    color: red;
}
.asc{
    background-color: yellow;}
</style>
<p class="desc intro asc">hi</p>
<p class="desc">hi</p>
<p class=“intro">hi</p>
```



Types of Selectors: **position in document**

```
<style>
   ul em {
         color: red;
          }
</style>
<body>
 <ul>
    <li>   <em>Hi</em>   </li>
 </ul>
    <p>    <em>Bye</em>  </p>
</body>
```



- Selects any em element that's a descendant of a ul element.
- The " " (space) is how you say "find a descendant."

Types of Selectors: **id + position**

- Selects any <li> element that is a descendant of any element with an id that equals “related-brands.”

```
<style>
   #related li {
                  color: blue;
                }
</style>
<body>
 <ul id="related">
    <li>Web </li>
    <li>Design </li>
 </ul>
</body>
```

Types of Selectors: **element + class**

```
<style>
li.special {
            color: red;
           }
</style>
<ul>
  <li>Web </li>
  <li class="special">Designing </li>
</ul>
```

- Selects any li element with a class attribute that contains the word "special".
- Warning: If you place a space “ ” after the “.”, it'll look for descendants of li with class of "special."

Types of Selectors: **All**

```
<style>
#header div.content form p em.required {
                              color: white;
            }
</style>
```

- Selects any em element with a class attribute that contains the word "required" that is a descendant of a p element that is a descendant of a form element that is a descendant of a div element with a class attribute that contains the word "content" that is a descendant of any element with an id attribute that equals "header".

CSS Grouping

Grouping Selectors

- You can group selectors to apply the same style to all of the selectors by separating them with commas:

```
<style>
a:hover, a:active:, a:focus {
background-color: yellow;
}
    </style>
```

```
h2 {
color: red;
}
.thisOtherClass {
color: red;
}
.yetAnotherClass {
color: red;
}
```

You can simply separate selectors with commas in one line and apply the same properties to them all so, making the above:

```
h2, .thisOtherClass, .yetAnotherClass {
color: red;
}
```

The selector: Cascade rules

```
<style>
#a #b h1 { color: red; } /* winner! */
    #a h1 { color: blue; }
</style>
```

- id is more specific than a class, class is more specific than element.
- the longer the selector, the more specific it is
- If style rules are equally specific, the last one wins!

Combining selectors

- Being able to define a CSS selector in terms of a tag, class or id is very powerful. But it's not practical to place classes on every tag in your document, much less to put unique ids throughout. It's also inconvenient to constantly repeat CSS rules. But by combining composing selectors all that can be avoided.

Comma separated selectors

- Let's say we want to make all our <blockquote> tags, <q> tags, and anything with "speech" in it's class string, to be red italic text. How might we do that? We could make three separate rule sets. Or, better, we can separate our selectors with commas (,) before one rule set.

Combining selectors

SEPERATE

```
<style>
blockquote{
color: red;
font-style: italic;
}
q{
color: red;
font-style: italic;
}
.speech{
color: red;
font-style: italic;
}
</style>
```

JOINED

```
<style>

blockquote,
q,
.speech {
    color: red;
    font-style:
    italic;
}
</style>
```

CSS Nesting

If the CSS is structured well, there shouldn't be a need to use many class or ID selectors. This is because you can specify properties to selectors within other selectors.

```
<style>
#top {
background-color: #ccc;
padding: 1em
}

#top h1 {
color: #ff0;
}

#top p {
color: red;
font-weight: bold;
}
</style>
```

This removes the need for classes or IDs on the p and h1 tags if it is applied to HTML that looks something like this:

```
<div id="top">
<h1>Chocolate curry</h1>
<p>This is my recipe for making curry purely with chocolate</
  p>
<p>Mmm mm mmmmm</p>
</div>
```

- This is because, by separating selectors with spaces, we are saying “h1 inside ID top is colour #ff0” and “p inside ID top is red and bold”.
- This can get quite complicated (because it can go for more than two levels, such as this inside this inside this inside this etc.) and may take a bit of practice.

Properties

PROPERTIES

Property Value Pairs

Each property can have one or more comma separated values.

- font: italic 12px sans-serif;
- color: #333;
- background-color: red;

Color

Properties : Color

The "color" property changes the text color. Colors can be specified either by name, for the most common colors, or by hexadecimal value.

- ✓ **color: red;**
- ✓ **color: #ff0000;**
- ✓ **color: rgb(255, 0, 0);**

This property is inherited, which means it'll also be applied to all descendant elements but can be overridden by more specific rules. This rule makes all the body text red unless specified otherwise:

```
<style>
   body {
   color: red;
   }
</style>
```

Properties : Color

```
b { color: transparent; }
    /* transparent */
```

- One of the more interesting is transparent, which will make the text invisible.
- However, if you want to make an HTML element invisible, then the **display:none; or visibility:hidden;** rules are preferred. They are discussed in a future section.

Properties: Background-Color

The "background-color" property changes the background color. Besides the BODY element, all elements default to a transparent background.

- ✓ background-color: black;
- ✓ background-color: #000000;
- ✓ background-color: rgb(0,0,0);

```
<style>
  body {
        background-color: yellow;
        }
  table {
        background-color: #FFCC00;
        }
</style>
```

Properties: Background-Image

- The background-image property sets one or more background images for an element.
- The background of an element is the total size of the element, including padding and border (but not the margin).

```
<style>
  body
     {
       background-image:url(‘paper.gif');
     }
</style>
```

Properties: Background-Repeat

The background-repeat property sets if/how a background image will be repeated.

```
background-repeat:  repeat|repeat-x|repeat-y|
  no-repeat|space|initial|inherit;
```

- **repeat-x** => The background image is repeated only horizontally
- **repeat-y** => The background image is repeated only vertically
- **space** => The background-image is repeated as much as possible without clipping. The first and last images are pinned to either side of the element, and whitespace is distributed evenly between the images.
- **round** => The background-image is repeated and squished or stretched to fill the space (no gaps).
- **initial** => Sets this property to its default value.
- **inherit** => Inherits this property from its parent element.

Properties: Background-Repeat

The background-repeat property sets if/how a background image will be repeated.

```
background-repeat: repeat|repeat-x|repeat-
  y|no-repeat|space|initial|inherit;
```

Repeat a background image both vertically and horizontally

```
<style>
body {
    background-image: url("paper.gif");
    background-repeat: repeat;
}
</style>
```

Properties: Background-Repeat

Repeat a background image only horizontally

```
<style>
body {
    background-image: url("paper.gif");
    background-repeat: repeat-x;
}
</style>
```

Do not repeat a background image. The image will only be shown once:

```
<style>
body {
    background-image: url("paper.gif");
    background-repeat: no-repeat;
}
</style>
```

Properties: background-size

```
background-size:  auto|length|cover|contain|initial|
  inherit;
```

- **auto** - Default value. The background image is displayed in its original size
- **Length**- Sets the width and height of the background image. The first value sets the width, the second value sets the height. If only one value is given, the second is set to "auto"
- **percentage** - Sets the width and height of the background image in percent of the parent element. The first value sets the **width**, the second value sets the **height**. If only one value is given, the second is set to “auto".
- **cover** - Resize the background image to cover the entire container, even if it has to stretch the image or cut a little bit off one of the edges
- **contain** - Resize the background image to make sure the image is fully visible.

Properties: background-size

```
<style>
#example1 {
    background: url(mountain.jpg);
    background-size: 100% 100%;
}

#example2 {
    background: url(mountain.jpg);
    background-repeat: no-repeat;
    background-size: 75% 50%;
}
</style>
```

Font

Properties : font-family

- The "font-family" property specifies the font family (or "font face") of the text.
- You can specify either a specific font name or a generic family name (serif, sans-serif, monospace, cursive).
  - ✓ font-family: "Times New Roman";
  - ✓ font-family: sans-serif;
- A comma separated list of font families can be specified if you want the browser to prefer one but use the others as backup options.

```
<style>
    p { font-family: "Helvetica", "Verdana",
      "Arial", sans-serif; }
</style>
```

Properties : font-size (em)

- The "em" unit lets you set the size of the text relative to the text around it.
- This makes the page resize nicely in proportion if the user changes their default font-size. The default size is "1em".

```
<style>
   p{
      font-size: 0.9em;
   }
   strong{
    font-size: 2.5em;
   }
</style>
<p>Hello </p>
<strong>output</strong>
```

Properties : font-size (px)

- The "px" unit lets you size font in terms of pixels, which is the unit also used to size images and other elements.
- It is easier to understand than em, but doesn't work as well when printing or resizing.

```
<style>
#hd2{
      font-size: 18px;
      }
</style>

<h2 id="hd2">Font 18px </h2>
<h2>Normal H2 Size</h2>
```

Font Size

Additionally, **`font-size`** supports a more readable set of values that many authors prefer: **`xx-small`**, **`x-small`**, **`small`**, **`medium`**, **`large`**, **`x-large`**, **`xx-large`** and relative sizing (relative to the text of the parent): **`larger`**, **`smaller`**. For example:

```
<style>
  p { font-size: medium; }
  #p2 { font-size: xx-small; }
  .p3 { font-size: xx-large; }
</style>
<p>welcome </p>
<p id="p2">welcome</p>
<p class="p3">welcome</p>
```

Properties : font-weight(bold)

- The "font-weight" property specifies the thickness of the font. The default is "normal" and the typical override is "bold".
- You can also specify "bolder", "lighter", or a number from 100 to 900.

```
<style>
  p { font-weight: bold; }
  blockquote { font-weight: 900; }
</style>
```

normal	bold	200
A Tale of Two Cities	**A Tale of Two Cities**	A Tale of Two Cities

500	700	900
A Tale of Two Cities	**A Tale of Two Cities**	**A Tale of Two Cities**

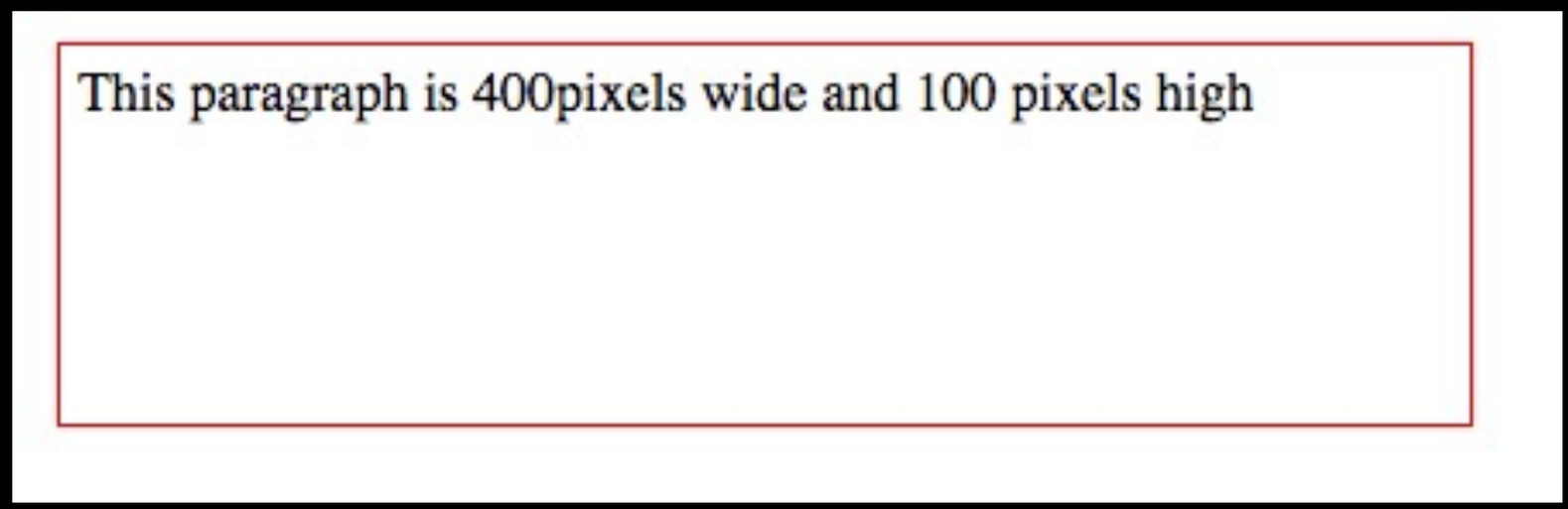
CSS - Dimension

CSS - Dimension

- We have seen the border that surrounds every box ie. element, the padding that can appear inside each box and the margin that can go around them. In this tutorial we will learn how we can change the dimensions of boxes.
- We have the following properties that allow you to control the dimensions of a box.
  - ✓ The **height** property is used to set the height of a box.
  - ✓ The **width** property is used to set the width of a box.
  - ✓ The **line-height** property is used to set the height of a line of text.
  - ✓ The **max-height** property is used to set a maximum height that a box can be.
  - ✓ The **min-height** property is used to set the minimum height that a box can be.
  - ✓ The **max-width** property is used to set the maximum width that a box can be.
  - ✓ The **min-width** property is used to set the minimum width that a box can be.

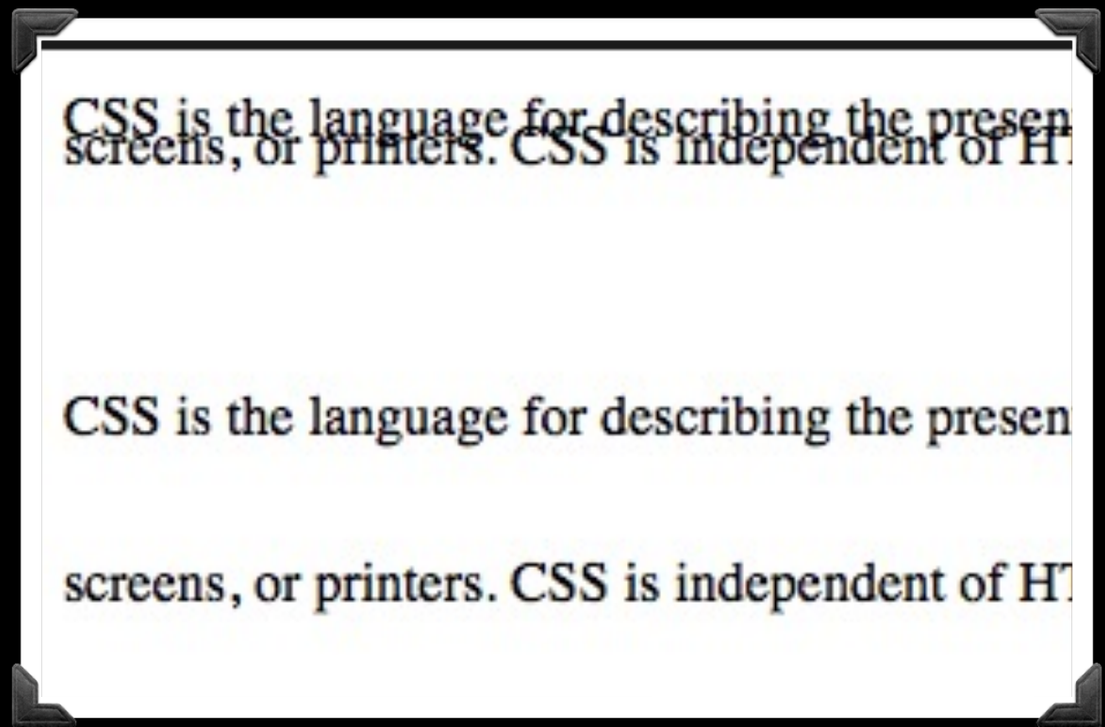
The Height and Width Properties

```
<body>
  <p style="width:400px; height:100px;
   border:1px solid red; padding:5px;
   margin:10px;">
This paragraph is 400 pixels wide and 100
   pixels high
</p>
</body>
```



Line Height

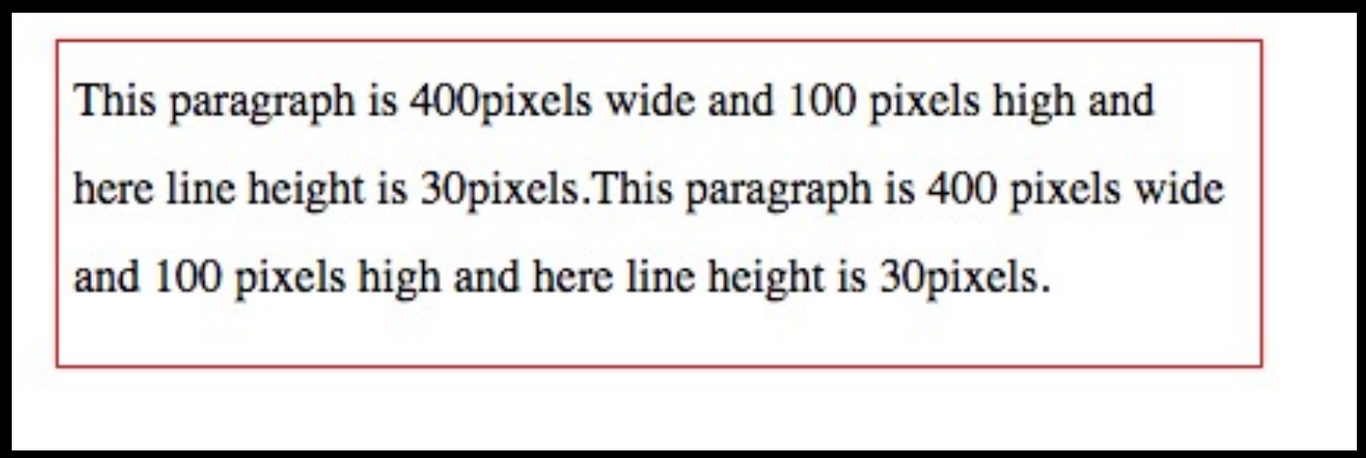
```
<style>
#p1 {
      line-height: 0.5;
     }
#p2 {
      line-height: 3.0;
    }
</style>
<p id="p1">CSS is the …… </p>
<br>
<p id="p2">CSS is the …… </p>
```



The line-height Property

The line-height property allows you to increase the space between lines of text. The value of the line-height property can be a number, a length, or a percentage.

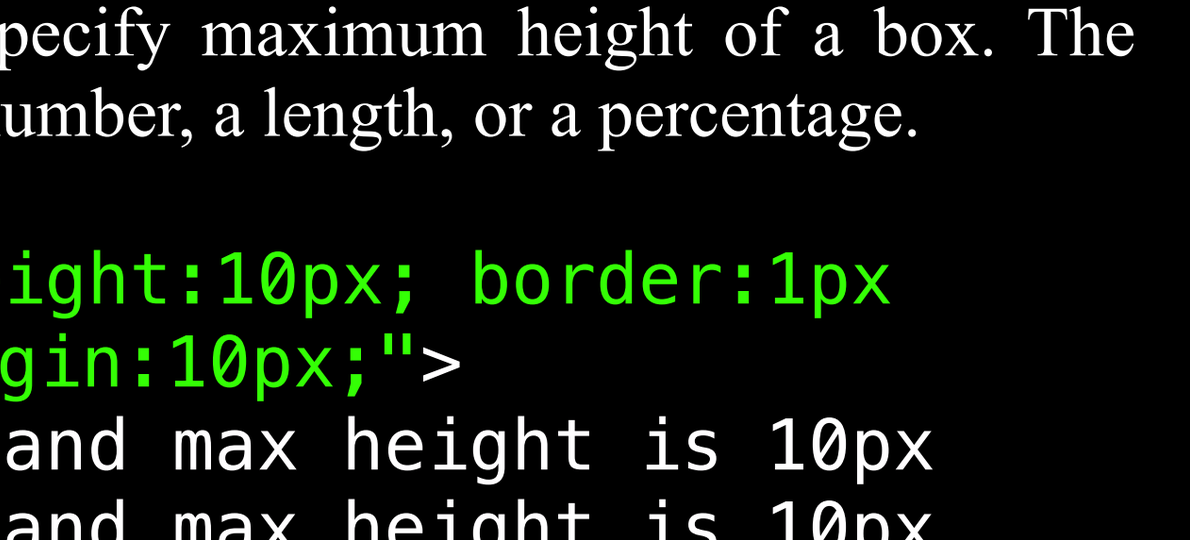
```
<body>
<p style="width:400px; height:100px; border:1px
      solid red; padding:5px; margin:10px; line-
      height:30px;">
This paragraph is 400pixels wide and 100 pixels high and here line height is 30pixels. This paragraph is 400 pixels wide and 100 pixels high and here line height is 30pixels.
</p>
</body>
```



The max-height Property

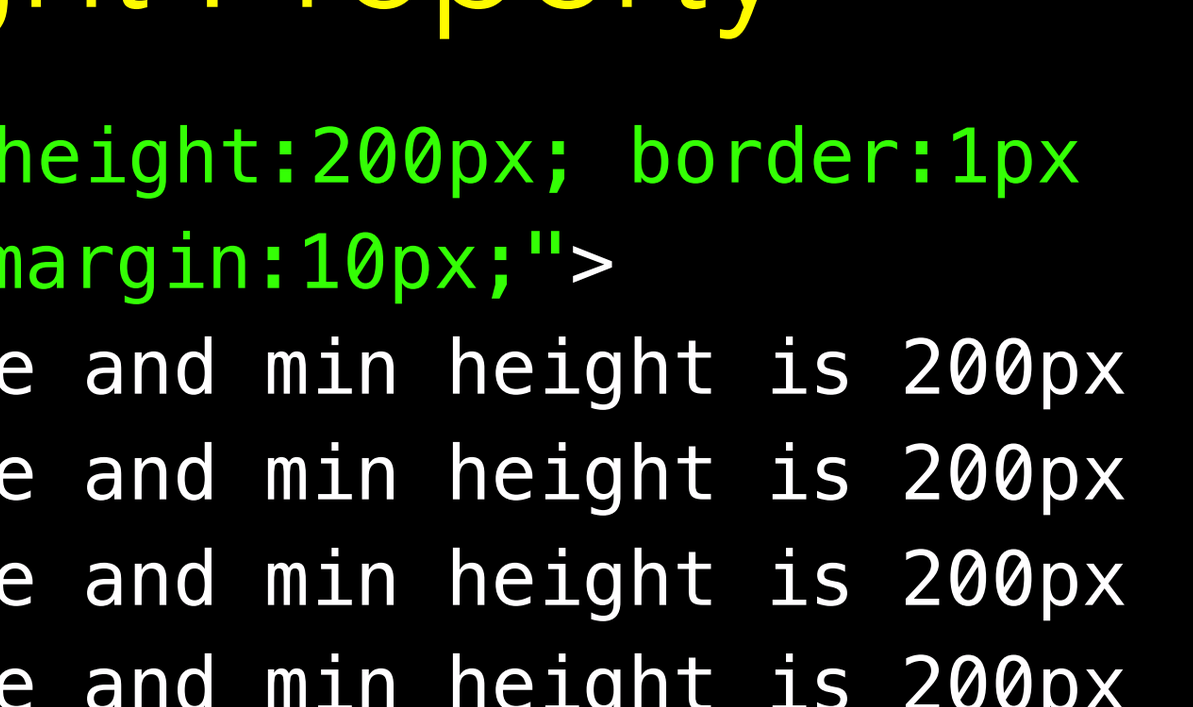
The max-height property allows you to specify maximum height of a box. The value of the max-height property can be a number, a length, or a percentage.

```
<body>
<p style="width:400px; max-height:10px; border:1px
  solid red; padding:5px; margin:10px;">
This paragraph is 400px wide and max height is 10px
This paragraph is 400px wide and max height is 10px
This paragraph is 400px wide and max height is 10px
This paragraph is 400px wide and max height is 10px
</p>
<br>   <br>   <br>
<img alt="logo" src="/css/images/logo.png" width="195"
  height="84" />
    </body>
```



The min-height Property

```
<p style="width:400px; min-height:200px; border:1px
  solid red; padding:5px;  margin:10px;">
This paragraph is 400px wide and min height is 200px
This paragraph is 400px wide and min height is 200px
This paragraph is 400px wide and min height is 200px
This paragraph is 400px wide and min height is 200px
</p>
<img alt="logo" src="/css/images/logo.png"
  width="95" height="84" />
</body>
```



Max & Min - Width

```
<style>
div {
    max-width: 500px;
    height: 100px;
    background-color: blue;
}
```

```
div {
    min-width: 200px;
    height: 100px;
    background-color: blue;
}
</style>
```

TEXT

Text Properties

- Anyone familiar with a text editor will be familiar with this property. Can be used to align the text **`left`**, **`center`** or **`right`**. There are additional possible values like **`justify`** and **`justify-all`** . Usually defaults to left. Remember that you shouldn't use `text-align` unnecessarily.
- Note that `text-align` may *not* work as expected if applied to elements that are the same width as their text, or whose width is determined by the text within them (i.e., inline elements).
- The tags **`<span>`**, **`<a>`**, **`<i>`**, **`<b>`**, **`<q>`** and others are considered "inline" because they do not receive their own new line when used. And `text-align` is often not useful on these tags.
- But it is useful on block level text tags, such as **`<p>`**, **`<li>`**, **`<ul>`**, **`<ol>`**, **`<div>`**, and **`<blockquote>`**

Text Properties

```
<style>
p { text-align: left; }
blockquote { text-align: right; }
</style>
```

Text-decoration

In CSS, this is done via the text-decoration property. The values for this are: **underline, overline, line-through, and none**; They can combined.

```
<style>
p { text-decoration: underline; }

a { text-decoration: none; }
/* hyperlinks are underlined by default,
   but that can be removed */

</style>
```

Text-decoration

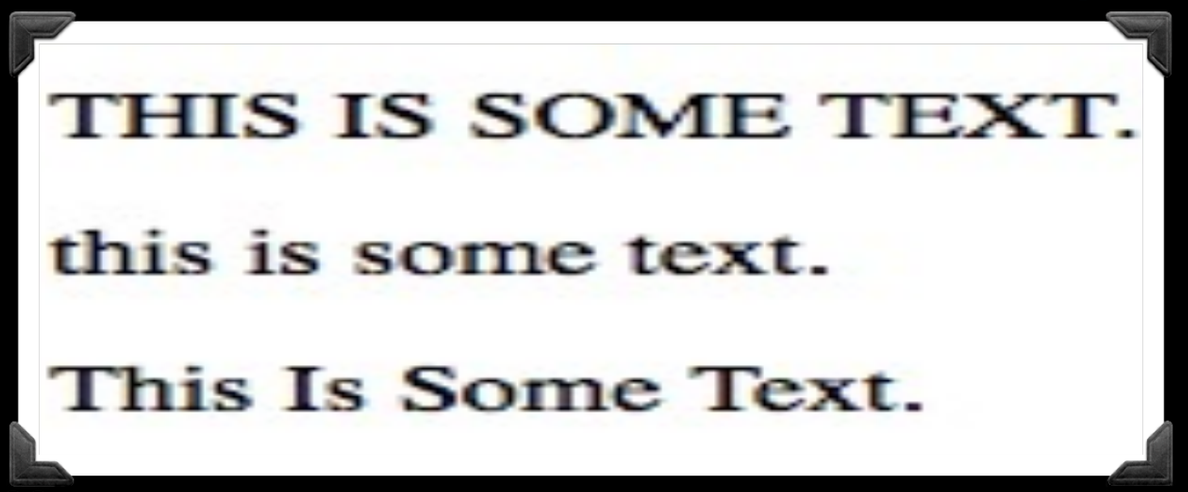
```
<style>
span { text-decoration: overline; }

span { text-decoration: underline
  overline; }
/* apply two with just a space between the
  values */

span { text-decoration: underline overline
  line-through; }       /* everything */
</style>
```

underline	overline	line-through	underline overline line-through
Middlemarch	Middlemarch	Middlemarch	Middlemarch

Text Tranformation

Text Transformation is used to specify uppercase and lowercase letters in a text.

```
<style>
p.uppercase { text-transform: uppercase;}
p.lowercase { text-transform: lowercase;}
p.capitalize { text-transform: capitalize;}
</style>
<body>
<p class="uppercase">This is some text.</p>
<p class="lowercase">This is some text.</p>
<p class="capitalize">This is some text.</p>
</body>
```



Text Indentation

Text Indentation is used to specify the indentation of the first line of a text.

```
<style>
   p {text-indent:50px;}
</style>
```

Styling Lists

Styling Lists -

- The list markup tags (<ul>, <ol> and <li>) are some of the most frequently used specific purpose tags in HTML. There are a few CSS style properties that are available for lists.
- **list-style-type** governs the little list marker that is usually positioned to the left of any list item. For un-ordered lists (<ul>) There are several popular values: disc, circle, square, and none.

```
li { list-style-type: disc; }
```

Styling Lists -

- For ordered lists (<ol>) you can choose different ways of having the numbers shown: decimal, decimal-leading-zero, lower-roman, upper-roman, lower-alpha, upper-alpha, as well as several of the worlds languages: armenian, georgian, simp-chinese-formal, and many others.

```
<style>
li { list-style-type: decimal-leading-
  zero; }
li { list-style-type: decimal; }
li { list-style-type: lower-roman; }
li { list-style-type: lower-alpha; }
</style>
```

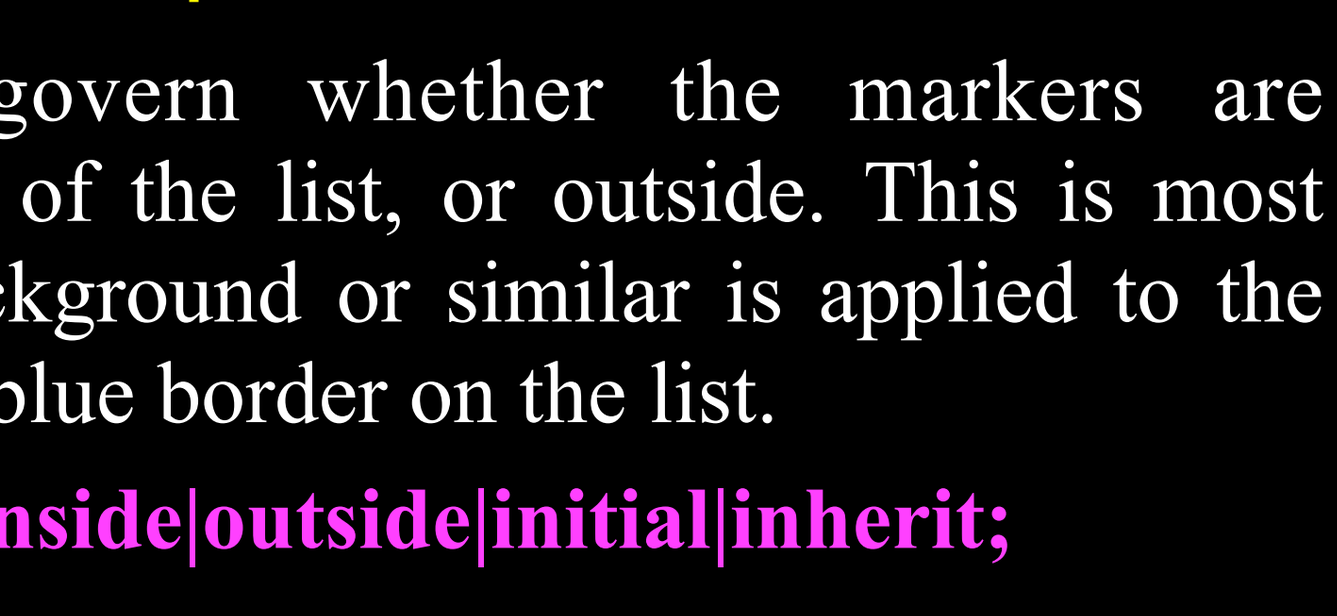
decimal	decimal-leading-zero	lower-roman	upper-alpha	simp-chinese-formal
1. eggs	01. eggs	i. eggs	A. eggs	壹、eggs
2. milk	02. milk	ii. milk	B. milk	贰、milk
3. bread	03. bread	iii. bread	C. bread	叁、bread

list-style-position

- The list-style-position govern whether the markers are positioned inside the box of the list, or outside. This is most evident if a border or background or similar is applied to the list. Below, we have put a blue border on the list.

list-style-position: inside|outside|initial|inherit;

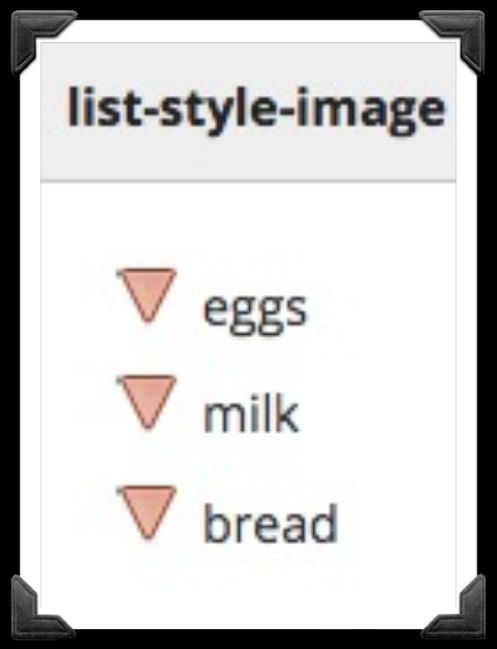
```
<style>
li{
    list-style-position: outside; border-
    color: blue; border:1px solid blue;
}
</style>
```



list-style-image

- The little markers on a list can also be customized to be an image of your choosing. This will require you to have a small image in a Web compatible format (PNG or JPEG recommended) and to know the path from the place where the CSS is being defined to the image.

```
li {
    list-style-image: url(“triangle.png");
   }
```



Styling Links

CSS Styling Links

The four links states are:

```
<style>
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
```

```
<style>
/* unvisited link */
a:link {
    color: red;
    background-color: yellow;
}
</style>
```

CSS Styling Links

```
<style>
/* visited link */
a:visited {
      color: red;
      background-color: yellow;
}
/* mouse over link */
a:hover{
      color: yellow;
            background-color: red;
}
/* active link*/
a:active {
      color: white;
      background-color: blue;
}
</style>
```

CSS Animations

CSS Animations

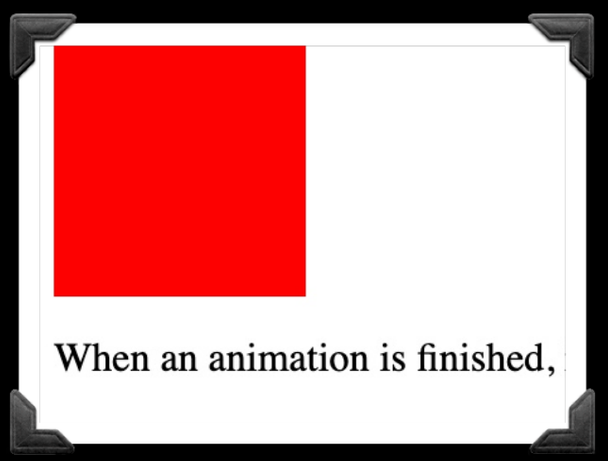
- CSS allows animation of HTML elements without using JavaScript or Flash!
- An animation lets an element gradually change from one style to another.
- When you specify CSS styles inside the **@keyframes** rule, the animation will gradually change from the current style to the new style at certain times.

CSS Animations

Property	Description
@keyframes	Specifies the animation code
animation	A shorthand property for setting all the animation properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long time an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies the name of the @keyframes animation
animation-play-state	Specifies whether the animation is running or paused
animation-timing-function	Specifies the speed curve of the animation

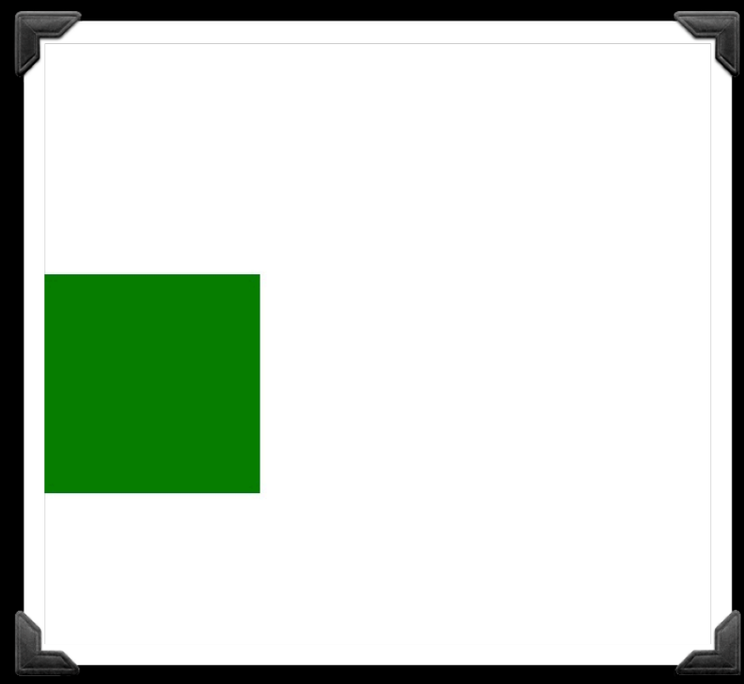
CSS Animations

```
<html> <head>
    <style>
      div {
        width: 100px;
        height: 100px;
        background-color: red;
        animation-name: example;
        animation-duration: 4s;
      }
      @keyframes example {
        from {background-color: red;}
        to {background-color: yellow;}
      }
  </style>  </head>
<body>
  <div></div>
  <p>When an animation is finished, it changes back to its original
     style.</p>
</body> </html>
```



CSS Animations

```
<style>
    div {
        width: 100px;
        height: 100px;
        background-color: red;
        position: relative;
        animation-name: example;
        animation-duration: 2s;
        animation-delay: 1s;
        animation-iteration-count: infinite;
    }
@keyframes example {
0%   {background-color:red; left:0px; top:0px;}
25%  {background-color:yellow; left:200px; top:0px;}
50%  {background-color:blue; left:200px; top:200px;}
75%  {background-color:green; left:0px; top:200px;}
100% {background-color:red; left:0px; top:0px;}
}
</style>
<body><div></div></body>
```



CSS Tables

Using CSS to Style Tables

CSS provides us with many tools to style our tables, again with the advantage of being able to make multiple changes to a page with a single edit in the <style> section of the document.

Style	Description
width	Width of table or row.
background-color	Background color of table, row or cell.
color	Color of text in table, row, or cell.
text-align	Text alignment in table, row or cell.
border	Border of table, row, or cell.
padding	Padding of table, row, or cell.

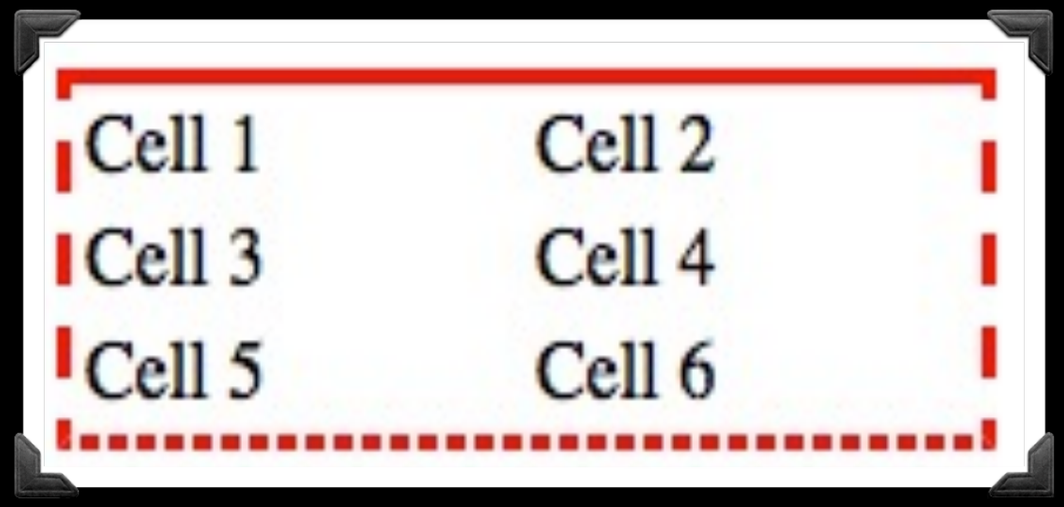
More About Borders

- The border-style and border-color properties can be defined for specific sides:
  - ✓ **Specifying 1 value** = style applies to all four sides.
  - ✓ **Specifying 2 value** = first applies to top and bottom, second applies to left and right.
  - ✓ **Specifying 3 value** = first applies to top, second applies to right and left, third applies to bottom.
  - ✓ **Specifying 4 value** = applies to top, right, bottom, left respectively.

More About Borders

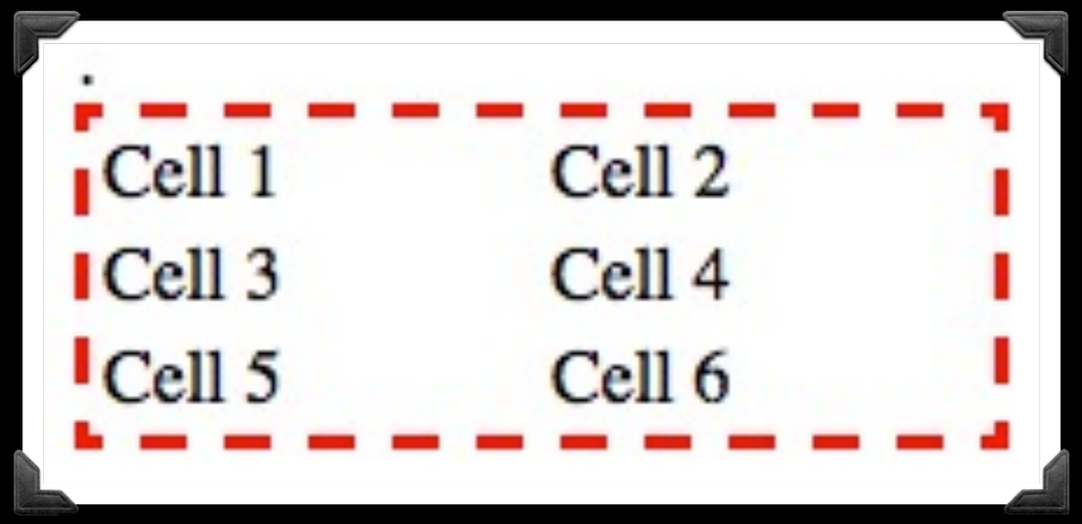
Style	Description
border-width	Width of border around table, row, or cell. Measured in pixels. Also available are: thin, medium, and thick. A value of "0" displays no border.
border-color	Color of border around table, row, or cell.
border-style	Type of border to display: solid, dashed, dotted, groove, ridge, inset, outset.
border-collapse	Collapses double lines around cells into one line.

```
<style type="text/css">
table {
    width:200px;
    border-width:3px;
    border-color:red;
    border-style:solid dashed dotted;
    }
</style>
```

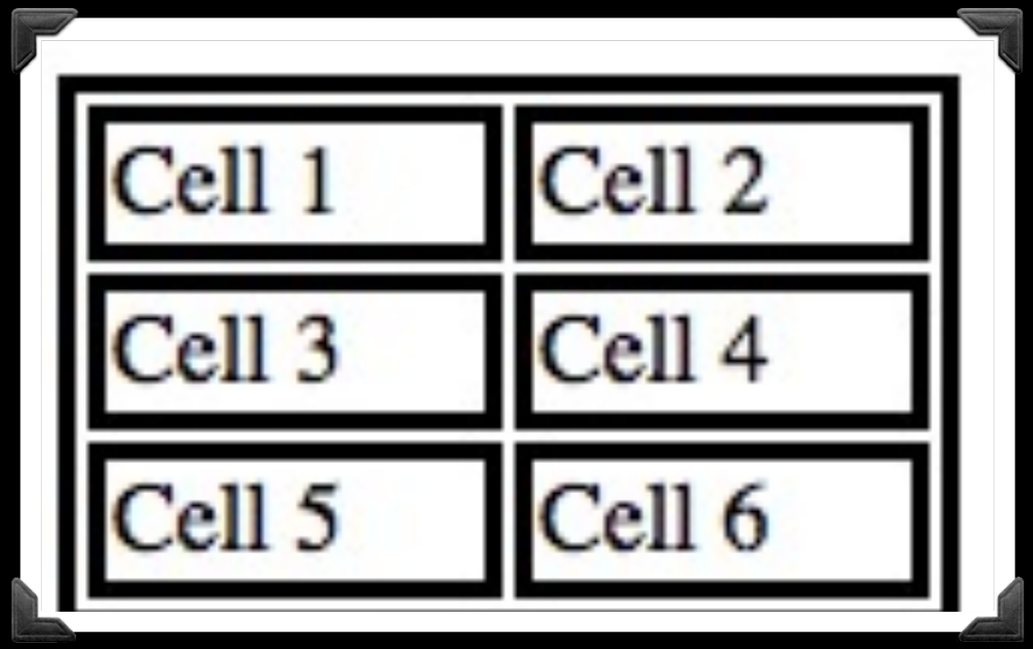



```
<table>
  <tr><td>Cell 1</td>. <td>Cell 2</td>    </tr>
  <tr><td>Cell 3</td><td>Cell 4</td>    </tr>
  <tr><td>Cell 5</td><td>Cell 6</td>    </tr>
</table>
</body>
```

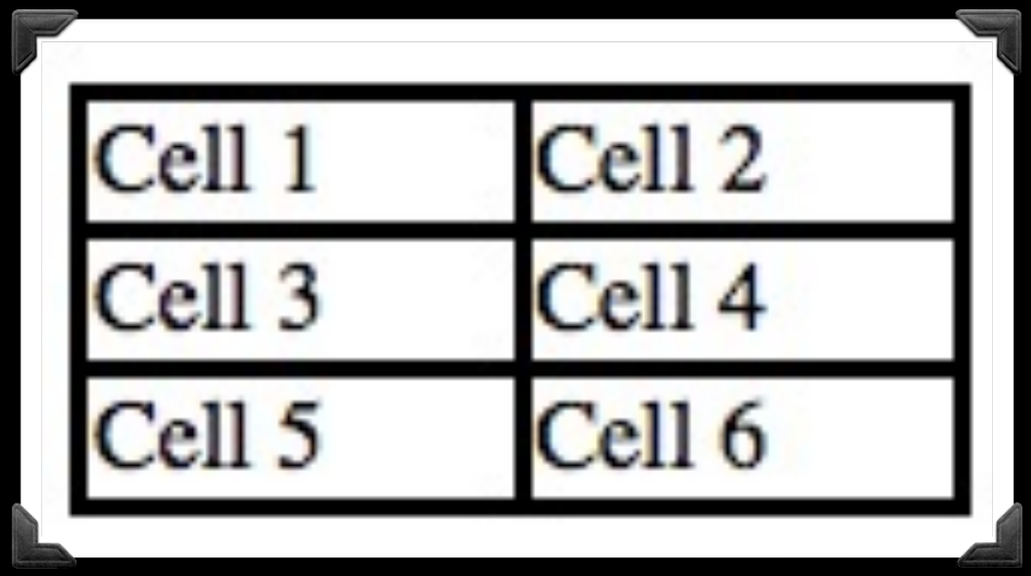
```
<style type="text/css">
table {
  width:200px;
  border:3px dashed red;
}
</style>
<table>
<tr><td>Cell 1</td>. <td>Cell 2</td>  </tr>
<tr><td>Cell 3</td><td>Cell 4</td>   </tr>
<tr><td>Cell 5</td><td>Cell 6</td>   </tr>
</table>
</body>
```



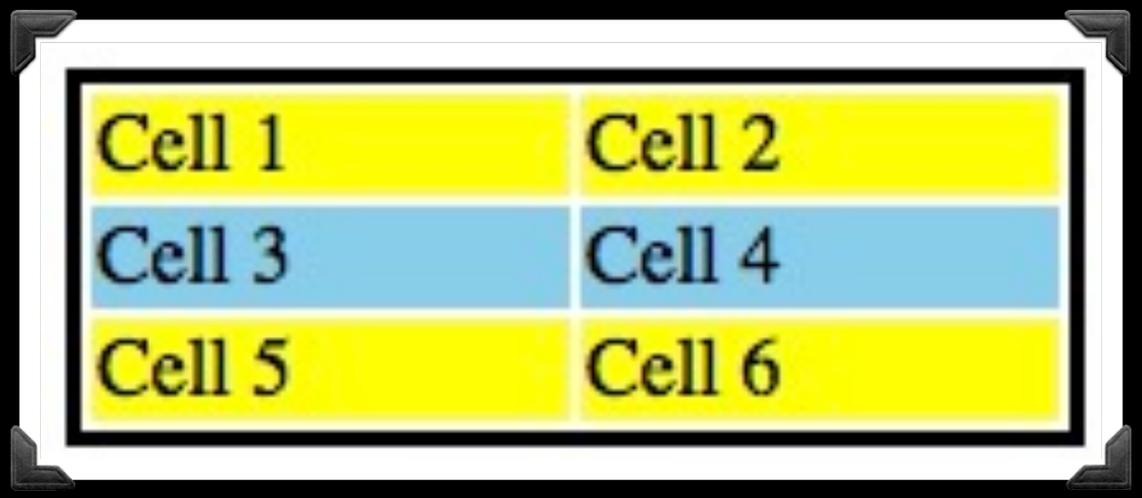
```
<style type="text/css">
table{
  width:150px;
  border:3px solid black;
}
tr td{
  border:3px solid black;  }
</style>
<table>
<tr><td>Cell 1</td><td>Cell 2</td>  </tr>
<tr><td>Cell 3</td><td>Cell 4</td>  </tr>
<tr><td>Cell 5</td><td>Cell 6</td>  </tr>
</table>
```



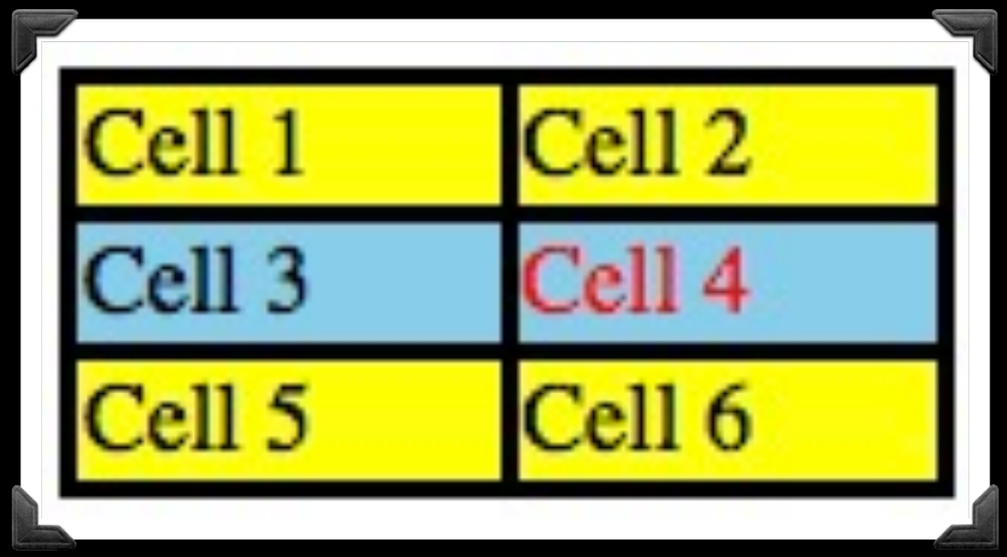
```
<style type="text/css">
table{
width:150px;
border:3px solid black;
border-collapse: collapse;
}
tr td{
border:3px solid black;  }
</style>
<table>
<tr><td>Cell 1</td><td>Cell 2</td>  </tr>
<tr><td>Cell 3</td><td>Cell 4</td>  </tr>
<tr><td>Cell 5</td><td>Cell 6</td>  </tr>
</table>
```



```
<style type="text/css">
   table {
   width:200px;
   border:3px solid black;
   }
   tr.odd {
   background-color:yellow;
   }
   tr.even {
   background-color:skyblue;
   }
</style>
<table>
<tr class="odd"><td>Cell 1</td><td>Cell 2</td></tr>
<tr class="even"><td>Cell 3</td><td>Cell 4</td></tr>
<tr class="odd"><td>Cell 5</td><td>Cell 6</td></tr>
</table>
```



```
<style type="text/css">
 table{
 width:150px;
 border:3px solid black;
 border-collapse: collapse;  }
 tr td{
    border:3px solid black;  }
 tr.odd {
 background-color:yellow;  }
 tr.even {
 background-color:skyblue;  }
 #new{
 color:red;}
</style>
<table>
<tr class="odd"><td>Cell 1</td><td>Cell 2</td></tr>
<tr class="even"><td>Cell 3</td><td id="new">Cell 4</td>
</tr><tr class="odd"><td>Cell 5</td><td>Cell 6</td></tr>
</table>
```



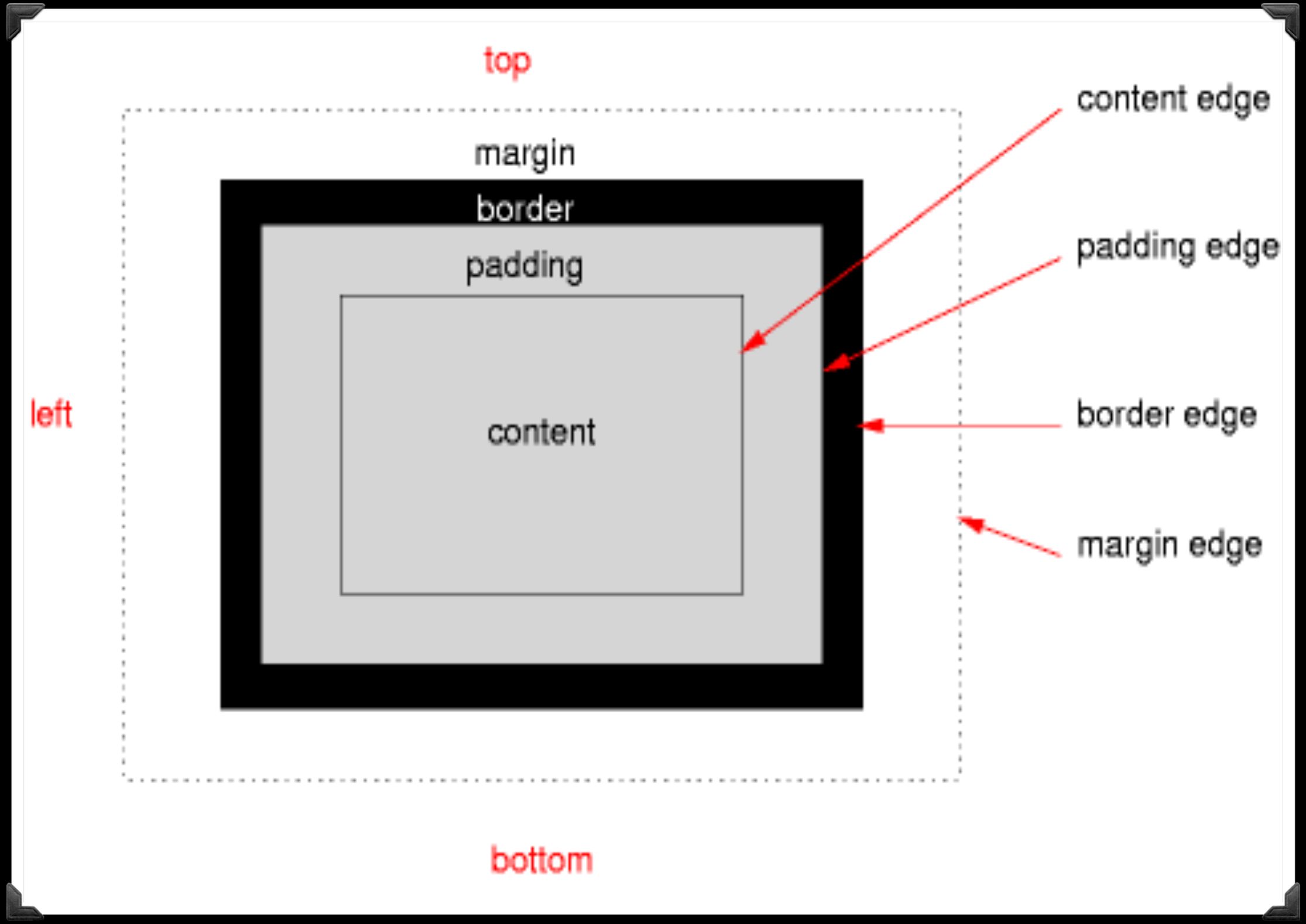
CSS Box Model

BOX Model

- The CSS box model describes the boxes that are formed around elements of content in a Web page.
- Every element in a Web page is in a box or is a box, even if it's an image of a circle.
- The boxes on Web pages are constrained by rules defined by the box model.

- The CSS box model is made up of four parts:
  - ✓ margin
  - ✓ border
  - ✓ padding
  - ✓ content

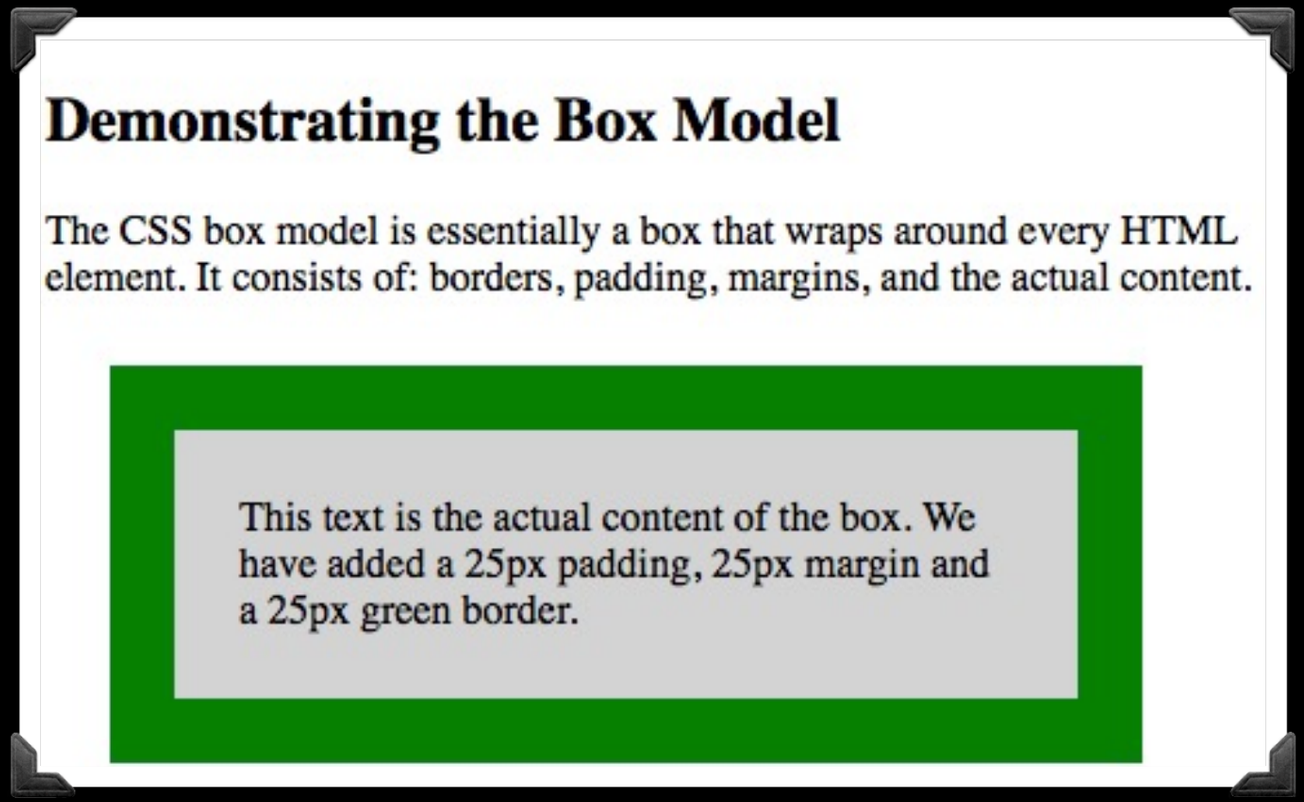
The CSS box model is made up of four parts



The CSS box model is made up of four parts

- Explanation of the different parts:
  - ✓ **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

```
<style>
div {
background-color: lightgrey;
width: 300px;
border: 25px solid green;
padding: 25px;
margin: 25px;
}
</style>
<h2>Demonstrating the Box Model</h2>
<p>The CSS box model is essentially a box that wraps
  around every HTML element. It consists of: borders,
  padding, margins, and the actual content.</p>
<div>This text is the actual content of the box. We
  have added a 25px padding, 25px margin and a 25px
  green border.</div>
```



```
<style>
div {
width: 350px;
padding: 10px;
border: 10px solid red;
margin: 0;
}
</style>
```

The picture above is 350px wide. The total width of this element is also 350px.

```
<h2>Calculate the total width:</h2>
<img src="flower.jpg" width="350" height="263">
<div>The picture above is 350px wide. The total
  width of this element is also 350px.</div>
```

```
<style>
div {
width: 320px;
padding: 10px;
border: 5px solid red;
margin: 0;
}
</style>
<h2>Calculate the total width:</h2>
<img src="flower.jpg" width="350" height="263">
<div>The picture above is 350px wide. The total
  width of this element is also 350px.</div>
```

320px (width) + 20px (left + right padding) + 10px (left + right border) + 0px (left + right margin)= 350px

Box Width and Height Calculation

- The total width of an element should be calculated like this:
  - ✓ Total element width = width + left padding + right padding + left border + right border + left margin + right margin
- The total height of an element should be calculated like this:
  - ✓ Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

Border

Border Style

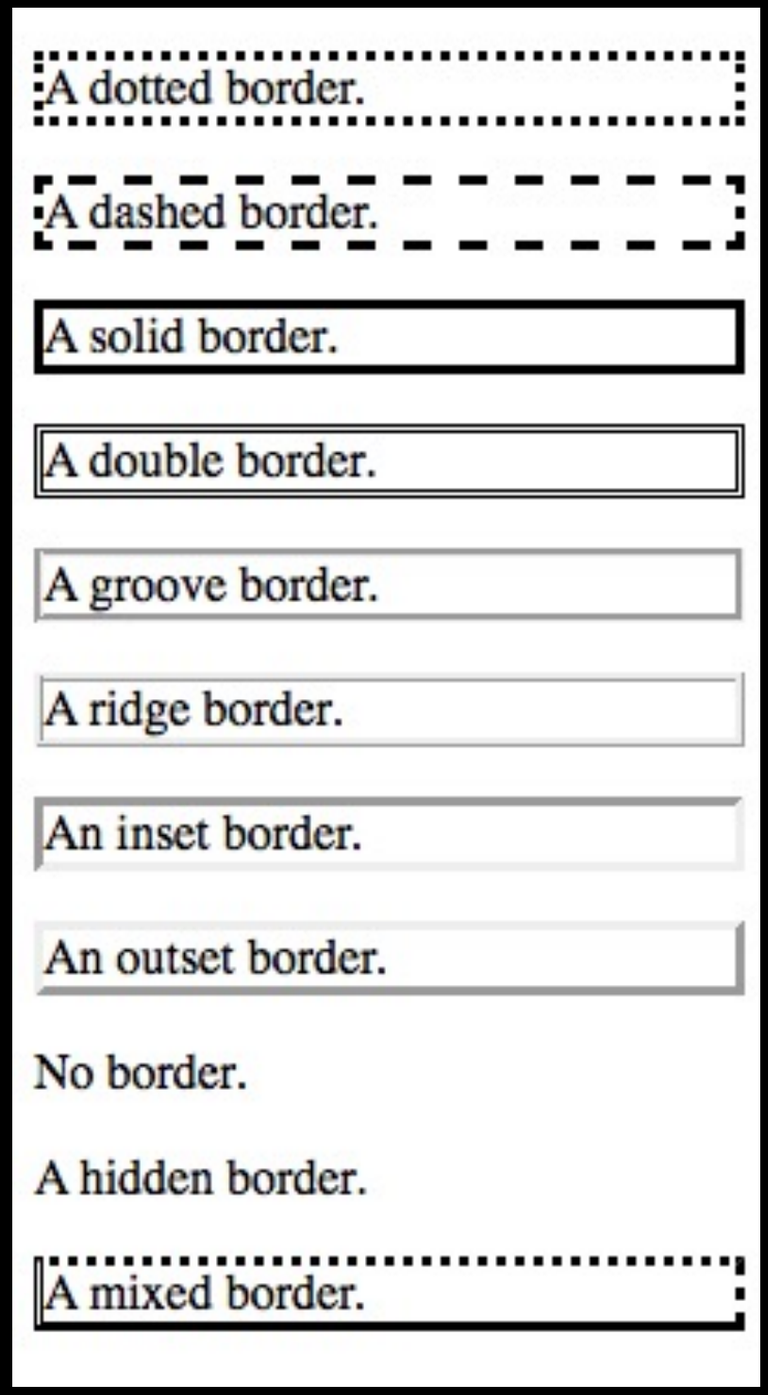
The border-style property specifies what kind of border to display.

The following values are allowed:

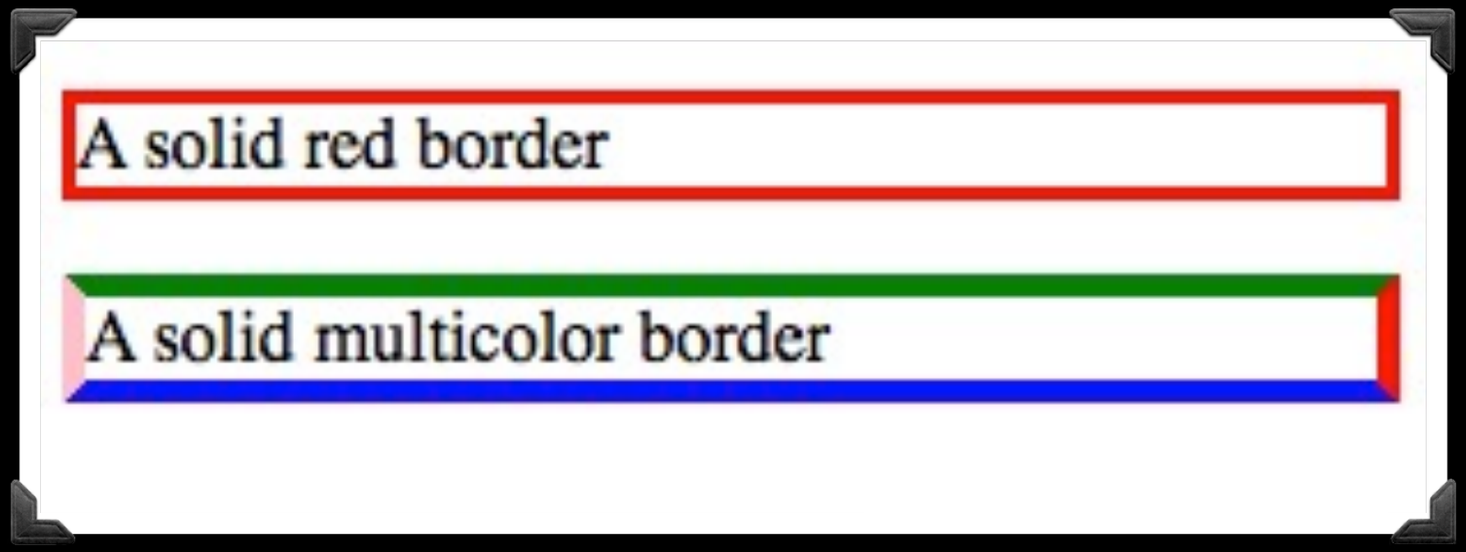
- ✓ dotted - Defines a dotted border
- ✓ dashed - Defines a dashed border
- ✓ solid - Defines a solid border
- ✓ double - Defines a double border
- ✓ groove - Defines a 3D grooved border. The effect depends on the border-color value
- ✓ ridge - Defines a 3D ridged border. The effect depends on the border-color value
- ✓ inset - Defines a 3D inset border. The effect depends on the border-color value
- ✓ outset - Defines a 3D outset border. The effect depends on the border-color value
- ✓ none - Defines no border
- ✓ hidden - Defines a hidden border

The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

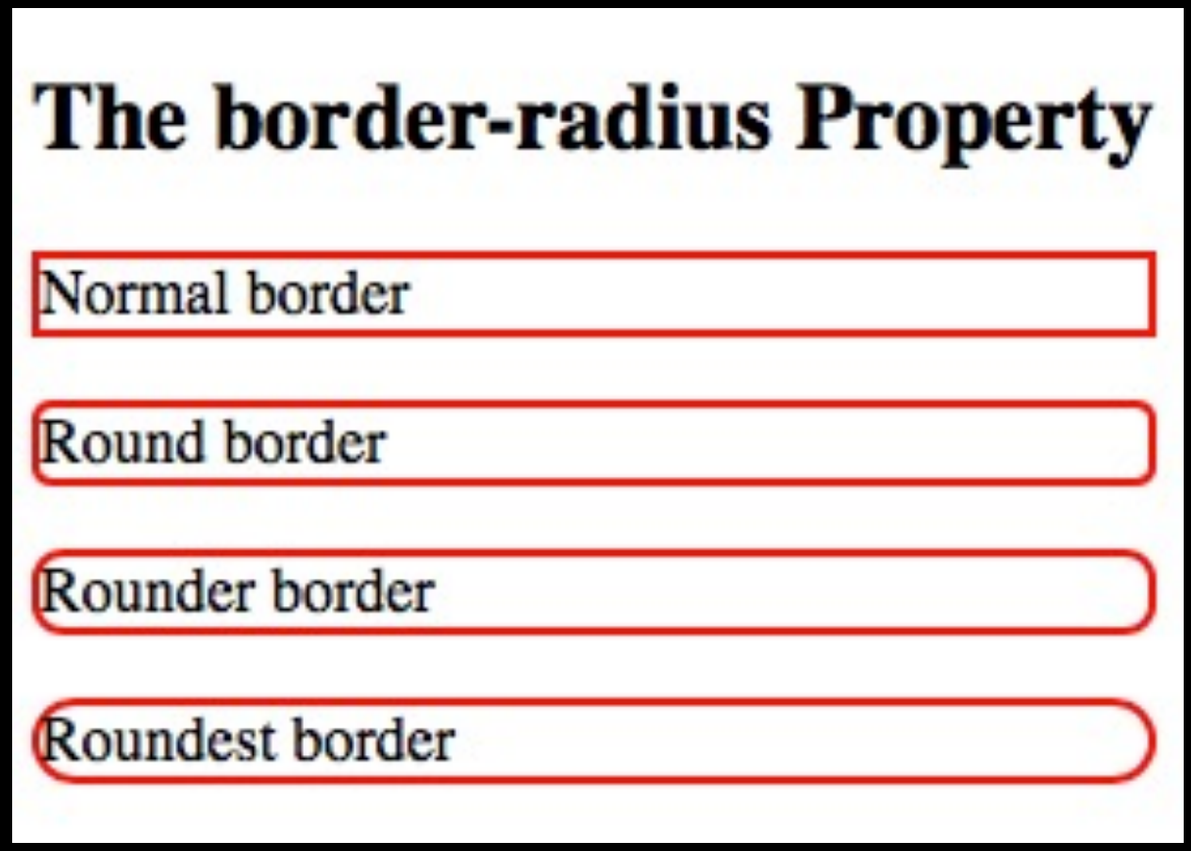
```
<style>
    p.dotted {border-style: dotted;}
    p.dashed {border-style: dashed;}
    p.solid {border-style: solid;}
    p.double {border-style: double;}
    p.groove {border-style: groove;}
    p.ridge {border-style: ridge;}
    p.inset {border-style: inset;}
    p.outset {border-style: outset;}
    p.none {border-style: none;}
    p.hidden {border-style: hidden;}
    p.mix {border-style: dotted dashed solid double;}
</style>
```



```
<style>
p.one {
border-style: solid;
border-color: red;
}
p.three {
border-style: solid;
border-color: green red blue pink;
border-width: 5px;
}
</style>
<p class="one">A solid red border</p>
<p class="three">A solid multicolor border</p>
```



```
<style>
p.normal {
    border: 2px solid red;}
p.round1 {
    border: 2px solid red;
    border-radius: 5px;  }
p.round2 {
    border: 2px solid red;
    border-radius: 8px;  }
p.round3 {
    border: 2px solid red;
    border-radius: 12px;  }
</style>
<h2>The border-radius Property</h2>
<p class="normal">Normal border</p>
<p class="round1">Round border</p>
<p class="round2">Rounder border</p>
<p class="round3">Roundest border</p>
```



Margin

Margin

- The margin property is the lynch-pin for positioning elements. Whenever you want to move something a little the margin property should be your first thought, when having layout problems, it is the first thing you should check.
- <u>The margin properties can have the following values:</u>
  - ✓ auto - the browser calculates the margin. Horizontally centered because it has margin.
  - ✓ *length* - specifies a margin in px, pt, cm, etc.
  - ✓ *%* - specifies a margin in % of the width of the containing element
  - ✓ inherit - specifies that the margin should be inherited from the parent element.

Tip: Negative values are allowed.

Margin

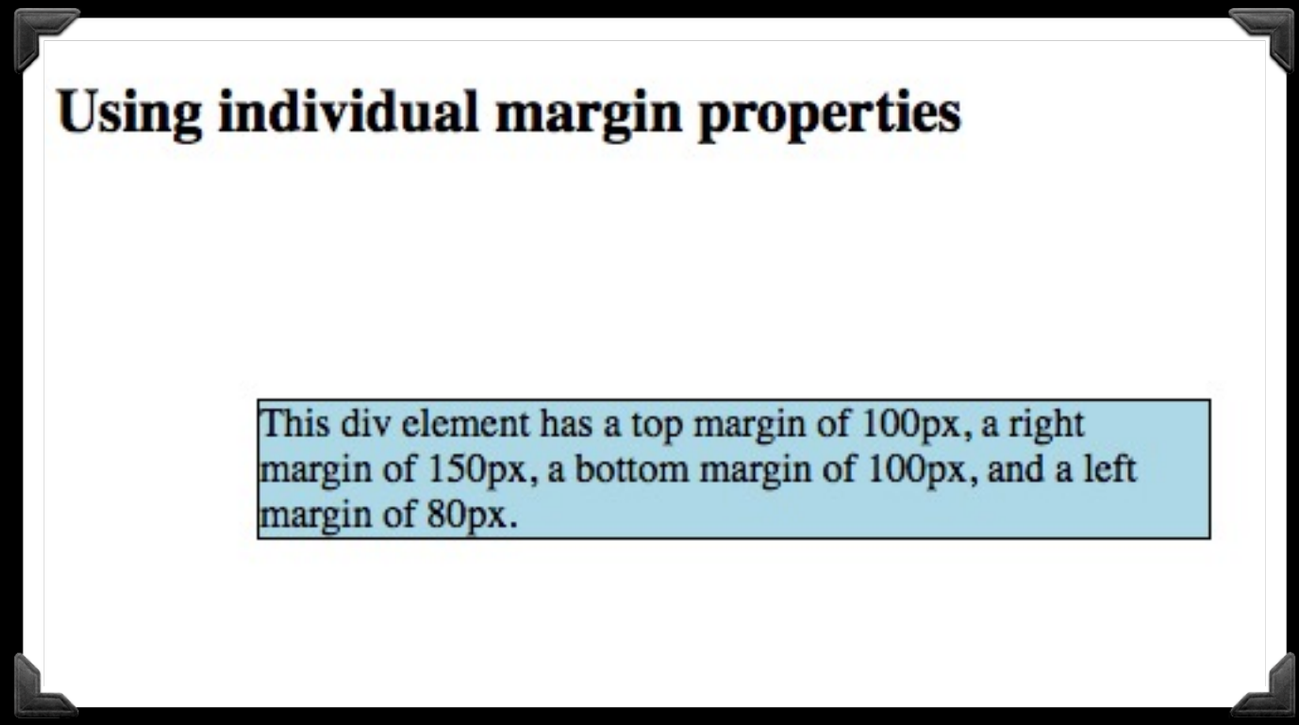
```
<style>
p { margin: 10px; } //Only ONE value – applied
   to all four sides.

◉ margin: 25px 50px 75px 100px;  // All Four
    Sides
◉ margin: 25px 50px 75px;  // Only THREE values
```

- ✓ Top margin is 25px
- ✓ Right and Left margins are 50px
- ✓ Bottom margin is 75px

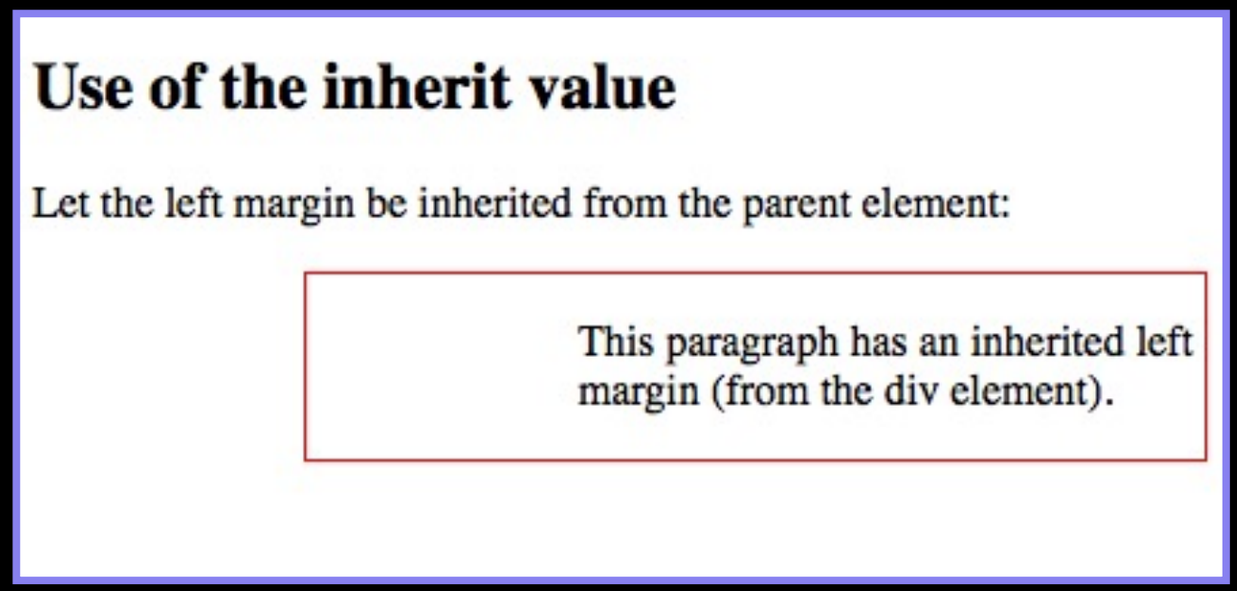
```
◉ margin: 25px 50px;  // Only TWO values
   }
```

- ✓ Top and Bottom margins are 25px
- ✓ Right and Left margins are 50px

```
<style>
div {
border: 1px solid black;
margin-top: 100px;
margin-bottom: 100px;
margin-right: 150px;
margin-left: 80px;
background-color: lightblue;
}
</style>
<h2>Using individual margin properties</h2>
<div>This div element has a top margin of 100px,
  a right margin of 150px, a bottom margin of
  100px, and a left margin of 80px.</div>
```



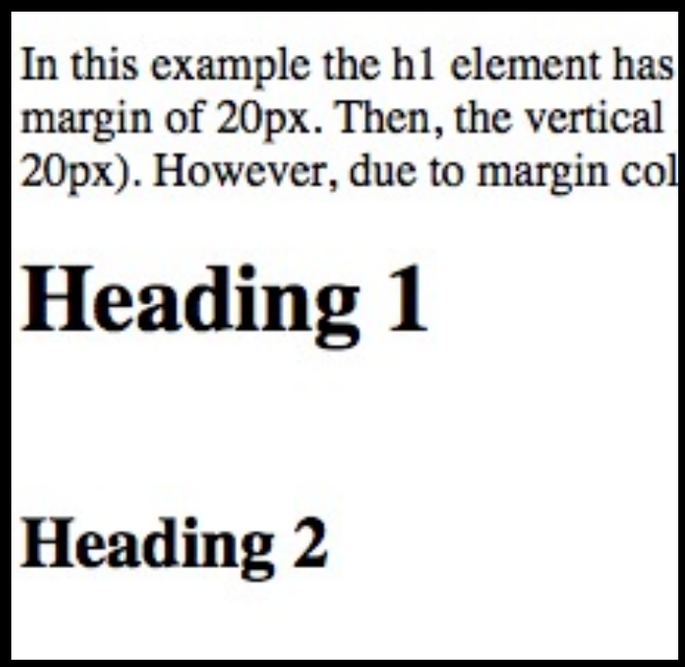
```
<style>
div {
    border: 1px solid red;
    margin-left: 100px;
}
p.ex1 {
    margin-left: inherit;
}
</style>
<h2>Use of the inherit value</h2>
<p>Let the left margin be inherited from the
  parent element:</p>
<div>
<p class="ex1">This paragraph has an inherited
  left margin (from the div element).</p>
</div>
```



Margin Collapse

- Top and bottom margins of elements are sometimes collapsed into a single margin that is equal to the largest of the two margins.
- This does not happen on left and right margins! Only top and bottom margins!

```
<style>
h1 { margin: 0 0 50px 0;  }
h2 { margin: 20px 0 0 0;   }
</style>
<p>In this example the h1 element has a bottom margin of 50px
  and the h2 element has a top margin of 20px. Then, the
  vertical margin between h1 and h2 should have been 70px
  (50px + 20px). However, due to margin collapse, the actual
  margin ends up being 50px.</p>
<h1>Heading 1</h1>
<h2>Heading 2</h2>
```

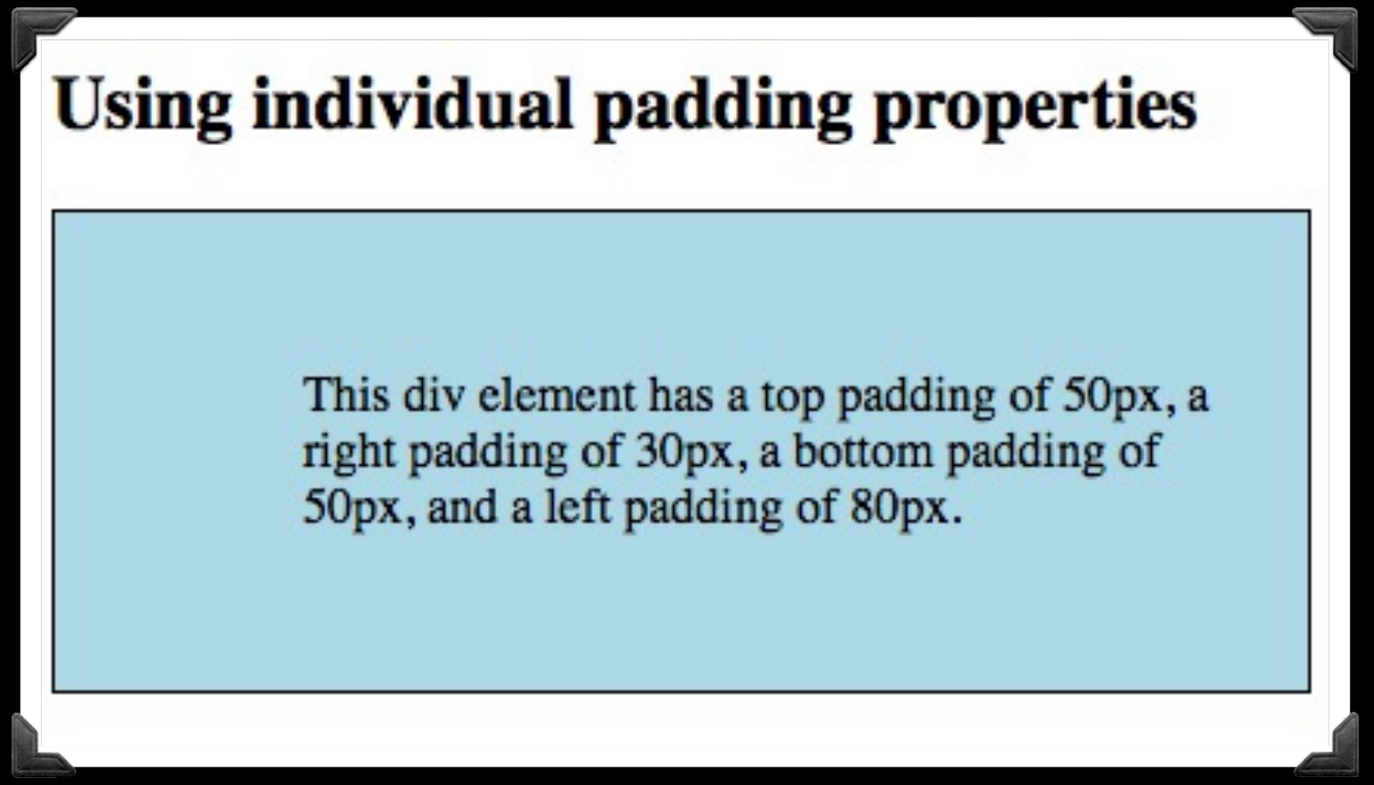


Padding

CSS Padding

- The CSS padding properties are used to generate space around an element's content, inside of any defined borders.
- CSS has properties for specifying the padding for each side of an element:
  - ✓ padding-top
  - ✓ padding-right
  - ✓ padding-bottom
  - ✓ padding-left
- All the padding properties can have the following values:
  - ✓ length - specifies a padding in px, pt, cm, etc.
  - ✓ % - specifies a padding in % of the width of the containing element
  - ✓ inherit - specifies that the padding should be inherited from the parent element

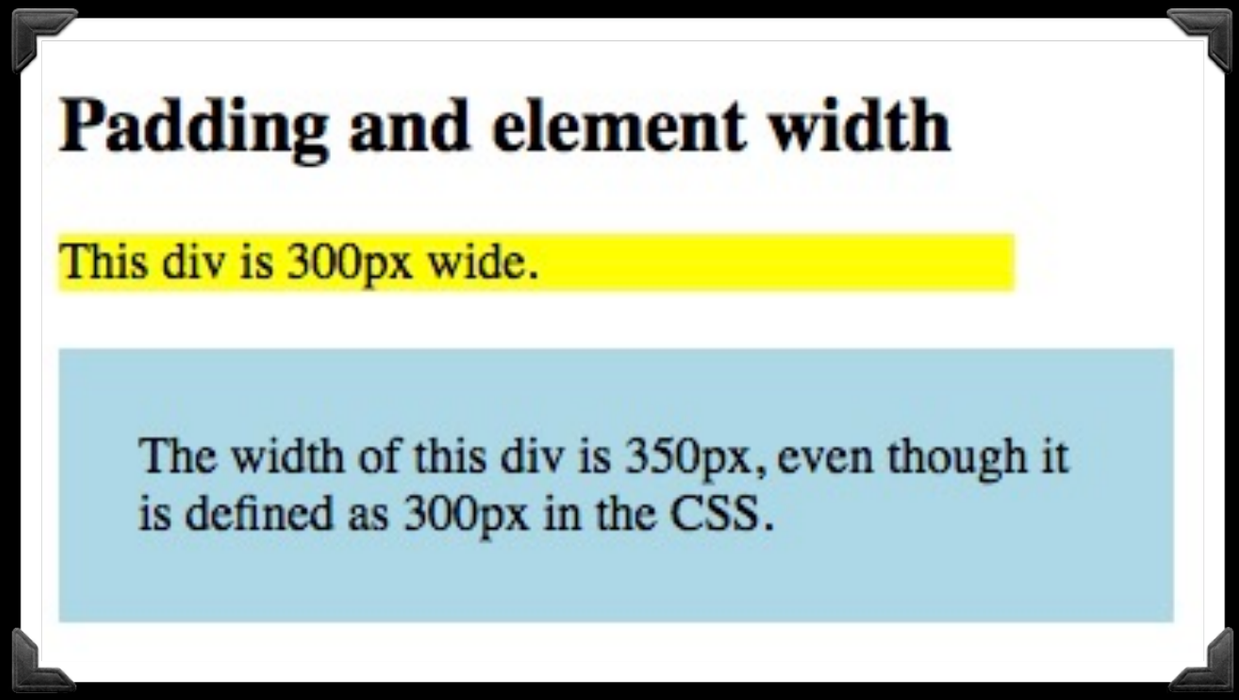
```
<style>
div {
border: 1px solid black;
background-color: lightblue;
padding-top: 50px;
padding-right: 30px;
padding-bottom: 50px;
padding-left: 80px;
}
</style>
```




```
<h2>Using individual padding properties</h2>
<div>This div element has a top padding of
  50px, a right padding of 30px, a bottom
  padding of 50px, and a left padding of
  80px.</div>
```

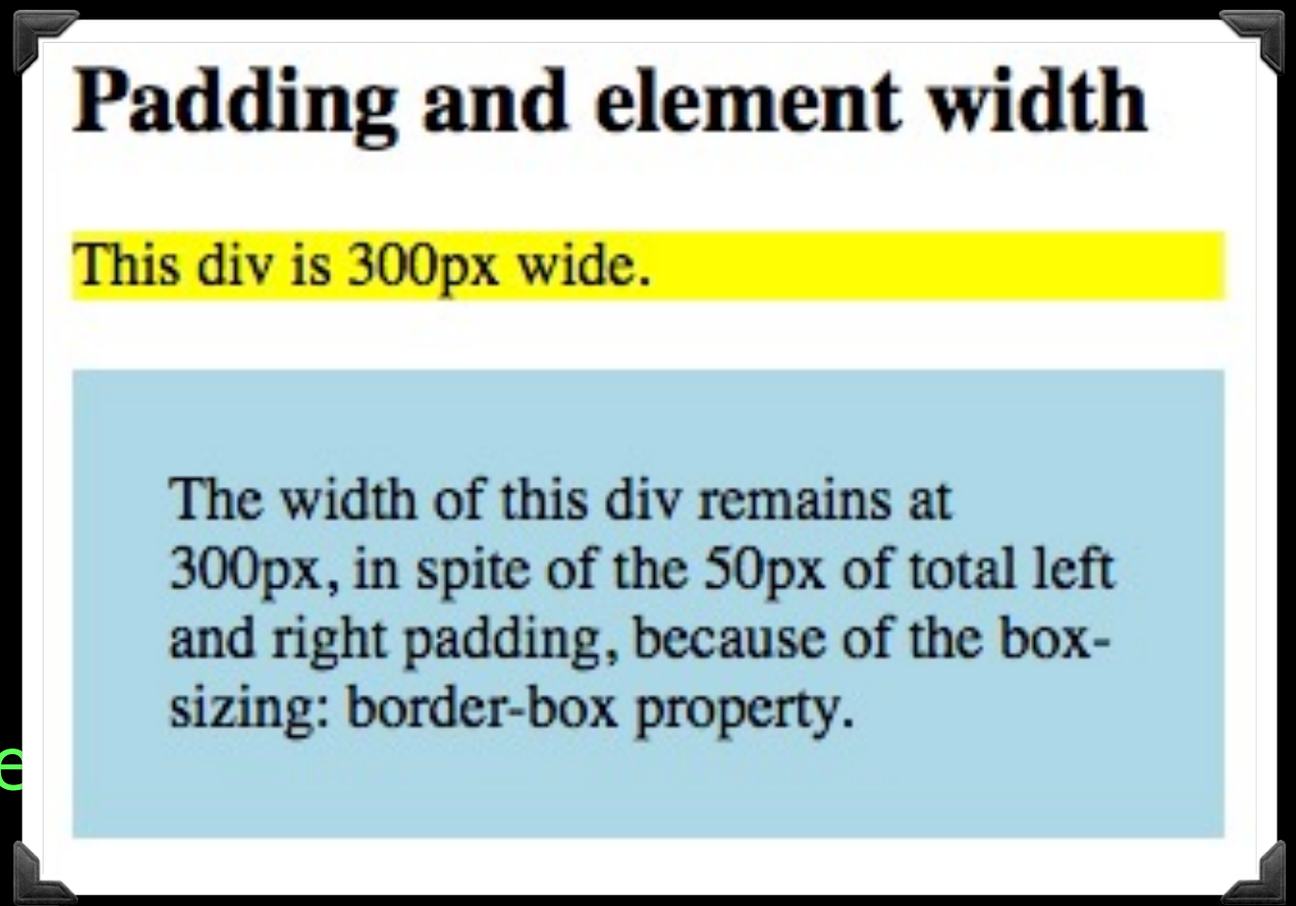
Padding and element width

```
<style>
div.ex1 {
  width: 300px;
  background-color: yellow;
}
div.ex2 {
    width: 300px;
    padding: 25px;
    background-color: lightblue;
}
</style>
<div class="ex1">This div is 300px wide.</div>
<br>
<div class="ex2">The width of this div is 350px, even
  though it is defined as 300px in the CSS.</div>
```



Padding and element width - Border Box

```
<style>
div.ex1 {
width: 300px;
background-color: yellow;
}
div.ex2 {
width: 300px;
padding: 25px;
box-sizing: border-box;
background-color: lightblue
}
</style>
<h2>Padding and element width</h2>
<div class="ex1">This div is 300px wide.</div>
<div class="ex2">The width of this div remains at
  300px, in spite of the 50px of total left and right
  padding, because of the box-sizing: border-box
  property. </div>
```



CSS Display

CSS Display

- Display is CSS's most important property for controlling layout.
- Every element has a default display value depending on what type of element it is.
  - ✓ The default for most elements is usually block or inline.
  - ✓ A block element is often called a block-level element.
  - ✓ An inline element is always just called an inline element.
- *CSS Syntax:* ***display: value;***

CSS Display Property Values

Value	Description
inline	Default value. Displays an element as an inline element (like). Any height and width properties will have no
block	Displays an element as a block element (like <p>). It starts on a new line, and takes up the whole width.
flex	Displays an element as a block-level flex 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-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

```
<style>
p {color: red;}
p.ex1 {display: none;}
p.ex2 {display: inline;}
p.ex3 {display: block;}
p.ex4 {display: inline-block;}
</style>
<h2>display: none:</h2>
<div>
Welcome to Web Development Class <p class="ex1">HELLO WORLD!</p> Have a
   great day!
</div>
<h2>display: inline (default):</h2>
<div>
Welcome to Web Development Class <p class="ex2">HELLO WORLD!</p> Have a
   great day!
</div>
<h2>display: block:</h2>
<div>
Welcome to Web Development Class <p class="ex3">HELLO WORLD!</p> Have a
   great day!
</div>
<h2>display: inline-block:</h2>
<div>
Welcome to Web Development Class <p class="ex4">HELLO WORLD!</p> Have a
   great day!
</div>
```

Output

display: none:

Welcome to Web Development Class Have a great day!

display: inline (default):

Welcome to Web Development Class HELLO WORLD! Have a great day!

display: block:

Welcome to Web Development Class

HELLO WORLD!

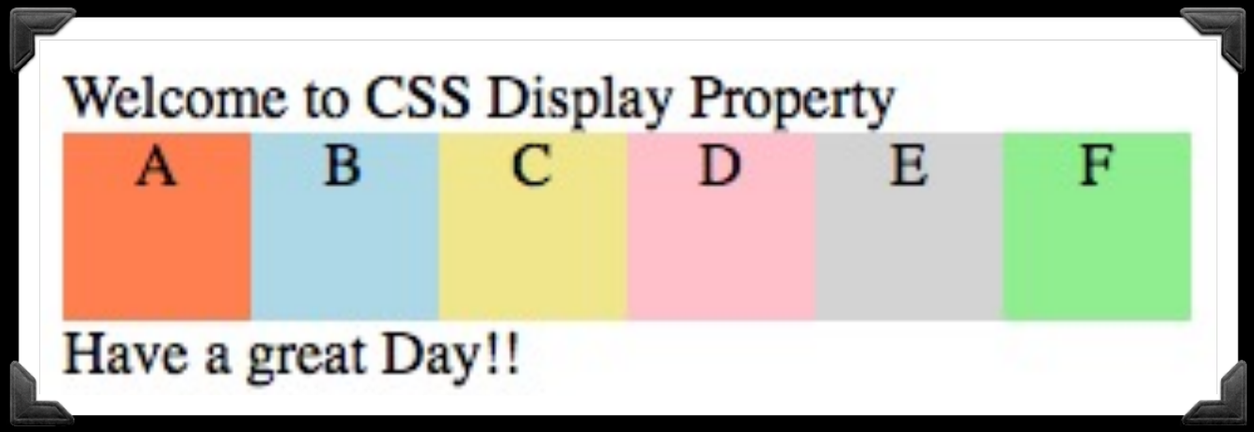
Have a great day!

display: inline-block:

Welcome to Web Development Class HELLO WORLD! Have a great day!

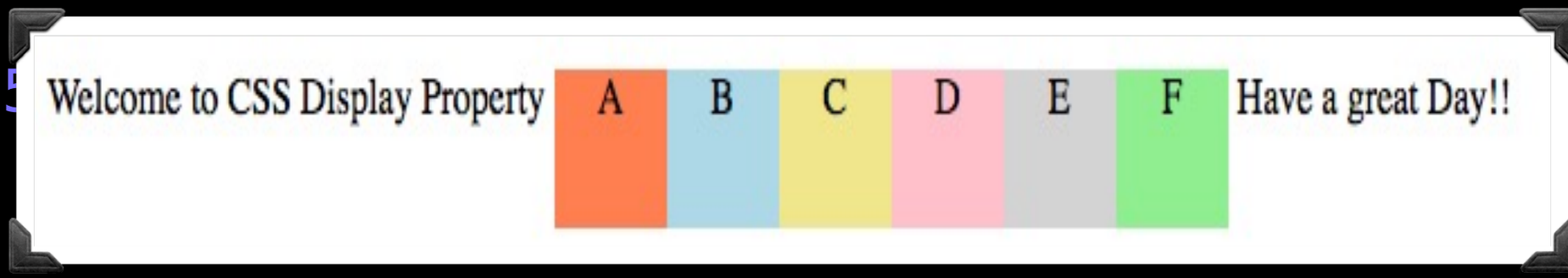
display: flex;

```
<style>
#main {
        display: flex;
        text-align: center;
}
#main div {
        width: 50px;
        height: 50px;
}
</style>
Welcome to CSS Display Property
<div id="main">
    <div style="background-color:coral;">A</div>
    <div style="background-color:lightblue;">B</div>
    <div style="background-color:khaki;">C</div>
    <div style="background-color:pink;">D</div>
    <div style="background-color:lightgrey;">E</div>
    <div style="background-color:lightgreen;">F</div>
</div>
Have a great Day!!
```



display: inline-flex;

```
<style>
#main {
        display: inline-flex;
        text-align: center;
}
#main div {
        width: 5
        height:
}
</style>
```




```
Welcome to CSS Display Property
<div id="main">
    <div style="background-color:coral;">A</div>
    <div style="background-color:lightblue;">B</div>
    <div style="background-color:khaki;">C</div>
    <div style="background-color:pink;">D</div>
    <div style="background-color:lightgrey;">E</div>
    <div style="background-color:lightgreen;">F</div>
</div>
Have a great Day!!
```

display: inline;

```
<style>
li {
  display: inline;
}
</style>
<p>Display a list of links as a horizontal
   menu:</p>
<ul>
<li><a href="/html.html">HTML</a></li>
<li><a href="/css.html">CSS</a></li>
<li><a href="/js.html">JavaScript</a>
</li>
</ul>
```

Display a list of links as a horizontal menu:

HTML CSS JavaScript

Note: inline changed the way of displaying unordered list

Display: none;

display: none; is commonly used with JavaScript to hide and show elements without deleting and recreating them.

For an example, let's use a **standard contact form**, like this one

If we set **display: none;** to that first input field, it would look like this:

```
<style>
#name{
display: none;
}
</style>
```

visibility: hidden;

Hiding an element can be done by setting the display property to none. The element will be hidden, and the page will be displayed as if the element is not there:

If we set the input field to **visibility: hidden**, it would look like this

```
<style>
#name{
visibility: hidden;
}
</style>
```

CSS Positioning

CSS Positioning

The position property specifies the type of positioning method used for an element (static, relative, absolute or fixed).

```
position: static|absolute|fixed|relative|
          sticky|initial|inherit;
```

CSS Positioning Property Values

Value	Description
static	Default value. Elements render in order, as they appear in the document flow
absolute	The element is positioned relative to its first positioned (not static) ancestor element
fixed	The element is positioned relative to the browser window
relative	The element is positioned relative to its normal position, so "left:20px" adds 20 pixels to the element's LEFT position
sticky	The element 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).

position: static;

```
<style>
div.static {
  position: static;
  border: 3px solid #73AD21;
}
</style>
<h2>position: static;</h2>
<p>An element with position: static; is not
  positioned in any special way; it is
  always positioned according to the normal
  flow of the page:</p>
<div class="static">
This div element has position: static;
</div>
```

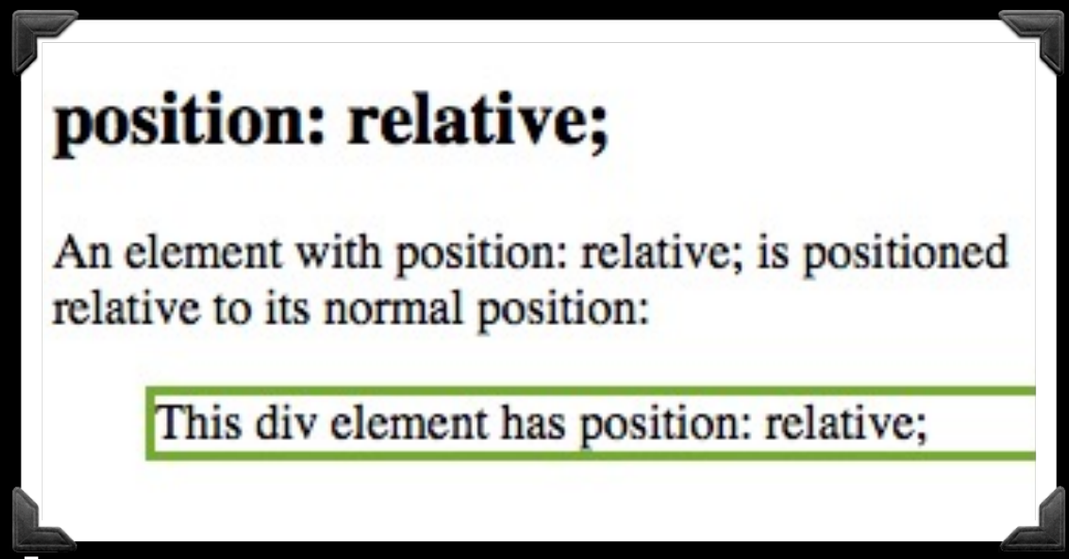
position: static;

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

This div element has position: static;

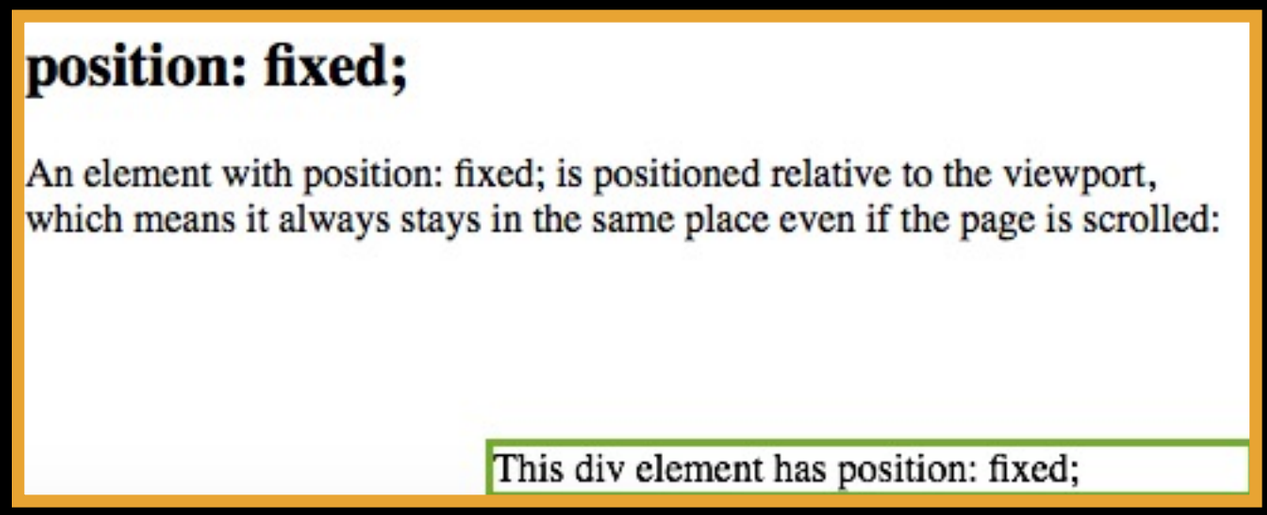
position: relative;

```
<style>
div.relative {
  position: relative;
  left: 30px;
border: 3px solid #73AD21;
}
</style>
<h2>position: relative;</h2>
<p>An element with position: relative; is
  positioned relative to its normal
  position:</p>
<div class="relative">
This div element has position: relative;
</div>
```



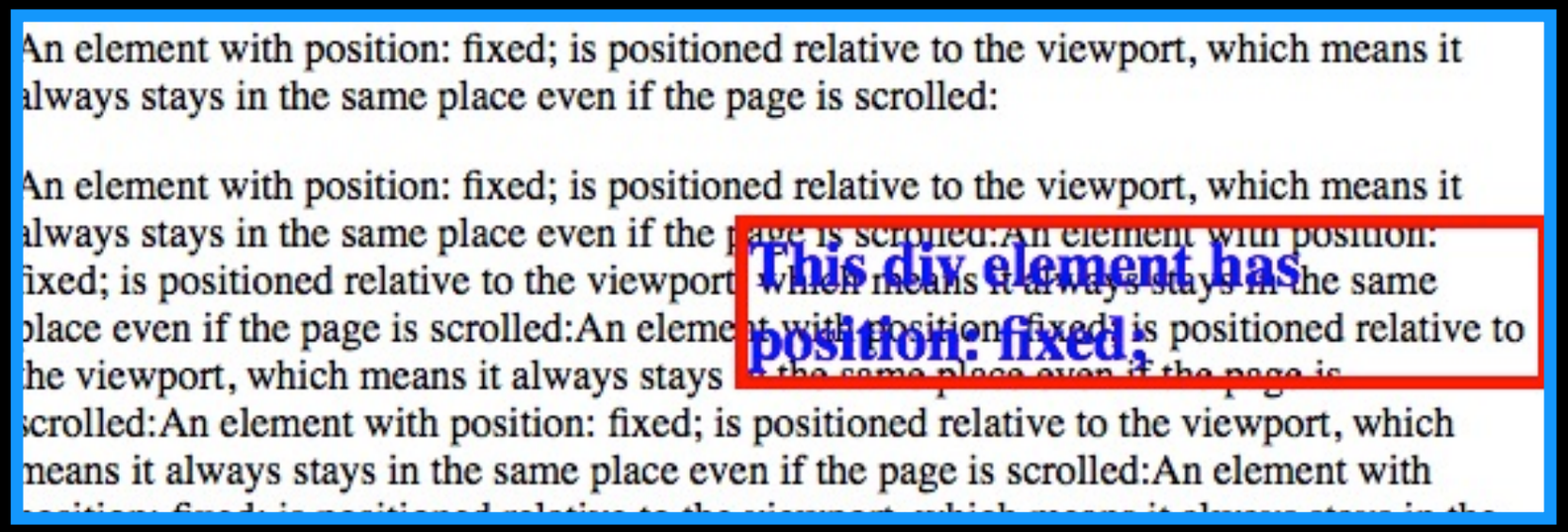
position: fixed;

```
<style>
div.fixed {
      position: fixed;
      bottom: 0;
      right: 0;
      width: 300px;
      border: 3px solid #73AD21;
}
</style>
<p>An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled:</p>
<div class="fixed">
This div element has position: fixed;
</div>
```



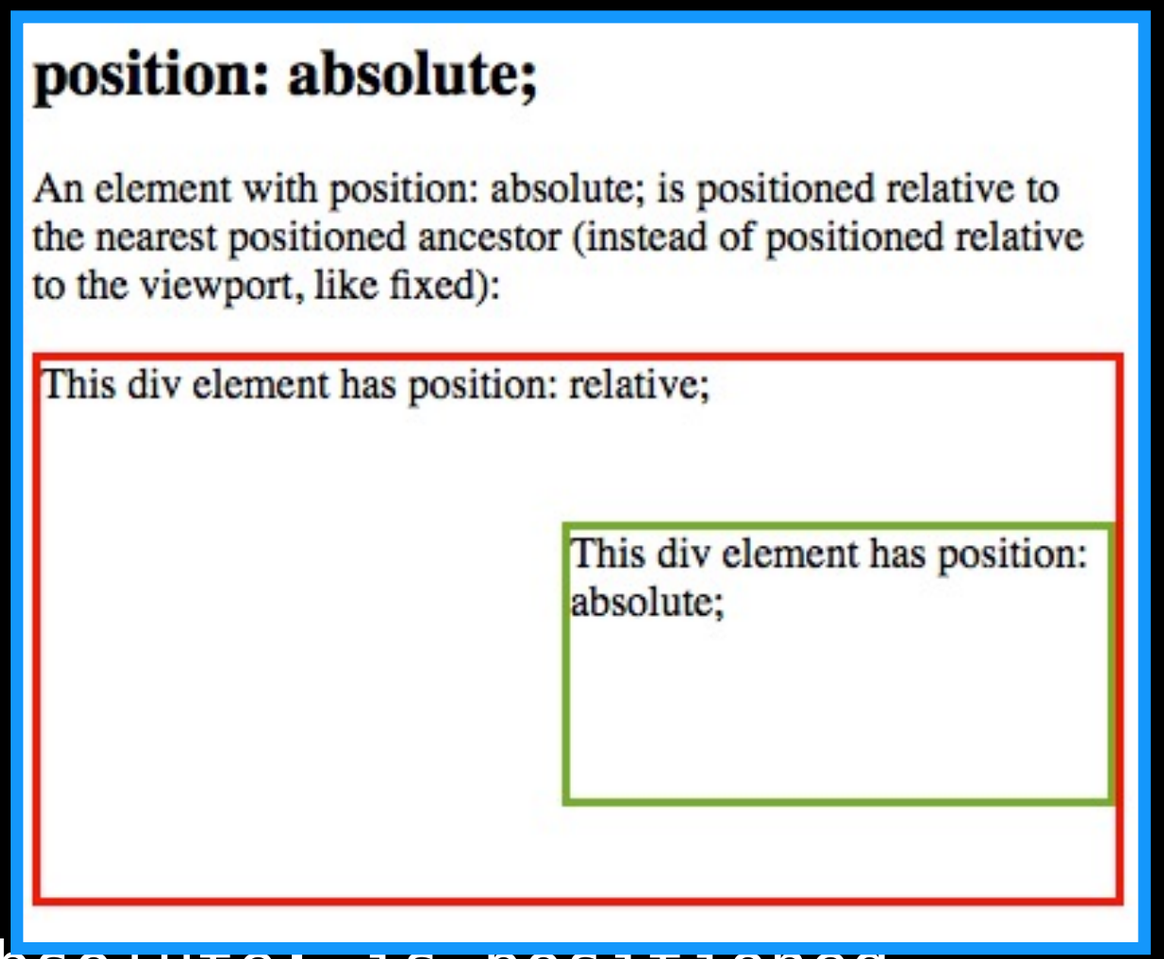
position: fixed;

```
<style>
h2.fixed {
 color: blue;
 position: fixed;
 right: 0;
 width: 300px;
 border: 5px solid red;
}
</style>
<p>An element with position: fixed; is positioned
   relative to the viewport, which means it always
   stays in the same place even if the page is
   scrolled:</p>
<h2 class="fixed">
This div element has position: fixed;</h2>
<p>An element with position: fixed…… </p>
```

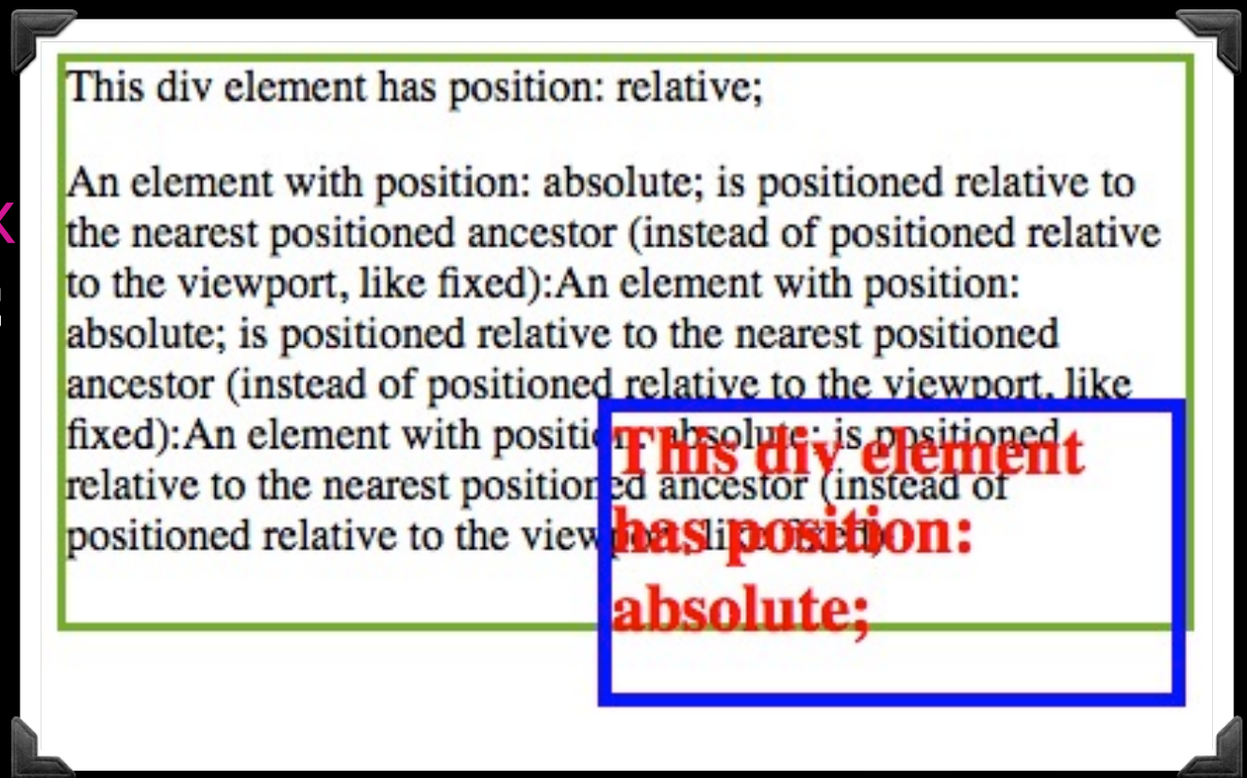


position: absolute;

```
<style>
div.relative {
position: relative;
width: 400px; height: 200px;
border: 3px solid red;
}
div.absolute {
position: absolute;
top: 60px; right: 0;
width: 200px; height: 100px;
border: 3px solid #73AD21;  }
</style>
<p>An element with position: absolute; is positioned
   relative to the nearest positioned ancestor (instead
   of positioned relative to the viewport, like fixed):</
   p>
<div class="relative">This div element has position:
   relative;
 <div class="absolute">This div element has position:
   absolute; </div></div>
```



```
<style>
div.relative {
position: relative;
width: 400px;height: 200px
border: 3px solid #73AD21;
}
h2.absolute {
position: absolute;
color:red;
top: 100px; right: 0;
width: 200px; height: 100px;
border: 5px solid blue;
}
</style>
<div class="relative">This div element has position:
    relative;
<h2 class="absolute">This div element has position:
        absolute;</h2>
<p>An element with position: absolute……</p>
</div>
```



position: sticky;

```
<style>
div.sticky {
      position: sticky;
      top: 0; padding: 5px;
      background-color: #cae8ca;
      border: 2px solid #4CAF50;
}
</style>
<p>Try to <b>scroll</b> inside this frame to
   understand how sticky positioning works.</p>
<div class="sticky">I am sticky!</div>
<div style="padding-bottom:2000px">
```

<p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position. Scroll back up to remove the stickyness……

Overlapping Elements

When elements are positioned, they can overlap other elements.

The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

```
<style>
img {
  position: absolute;
  left: 0px; top: 0px;
  z-index: -1;
}
</style>
<h1>This is a heading</h1>
<img src="css.gif" width="100" height="140">
<p>Because the image has a z-index of -1, it
   will be placed behind the text.</p>
```

Overlapping Elements

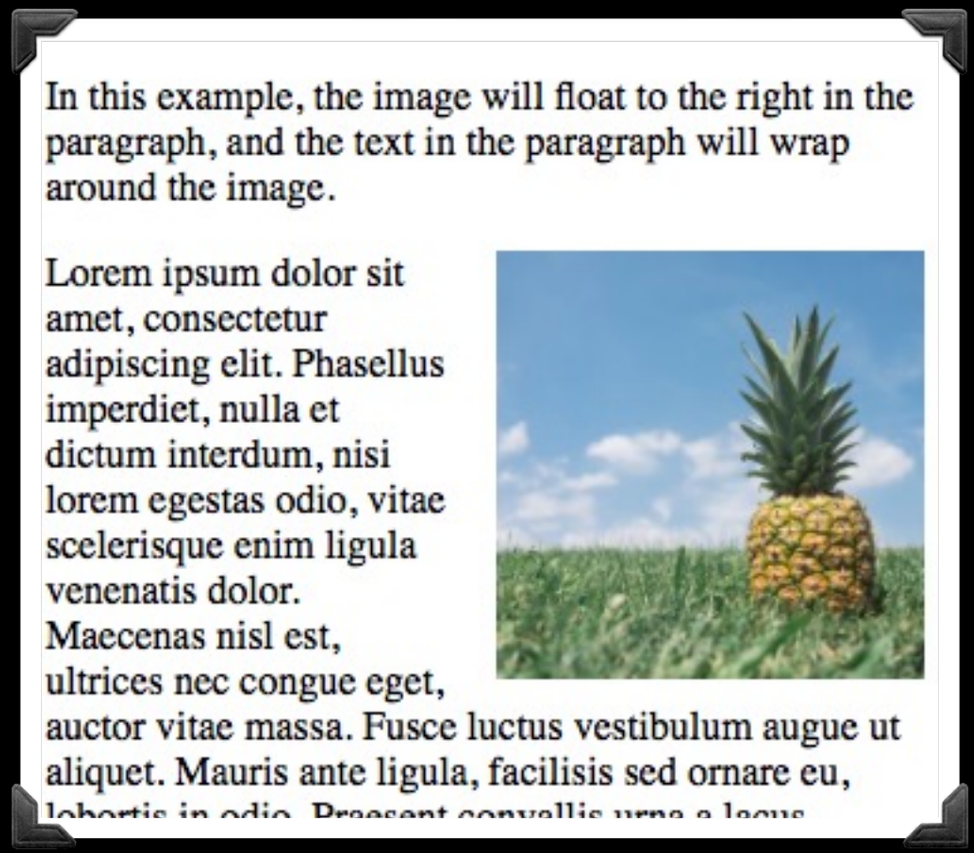
```
<style>
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: 1;
}
</style>
<h1>This is a heading</h1>
<img src="css.gif" width="100" height="140">
<p>Because the image has a z-index of 1, it
   will be placed front the text.</p>
```

CSS Float & Clear

CSS Float

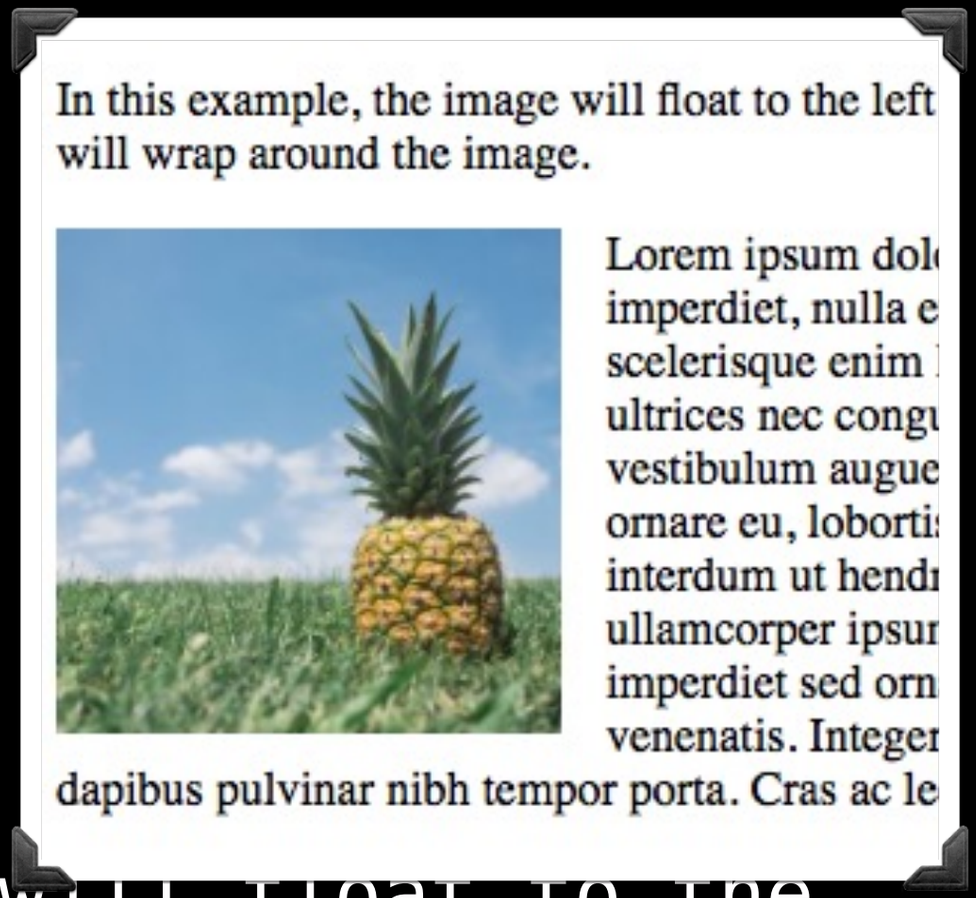
- The CSS float property specifies how an element should float.
- The float property is used for positioning and layout on web pages.
- The float property can have one of the following values:
  - ✓ left - The element floats to the left of its container
  - ✓ right- The element floats to the right of 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

float:right;




```
<style>
img {
      float: right;
}
</style>
<p>
In this example, the image will float to the
   right in the paragraph, and the text in the
   paragraph will wrap around the image.</p>
<p><img src="pineapple.jpg"
   style="width:170px;height:170px;margin-
   left:15px;">
Lorem ipsum dolor sit amet, consectetur
   adipiscing elit……
```

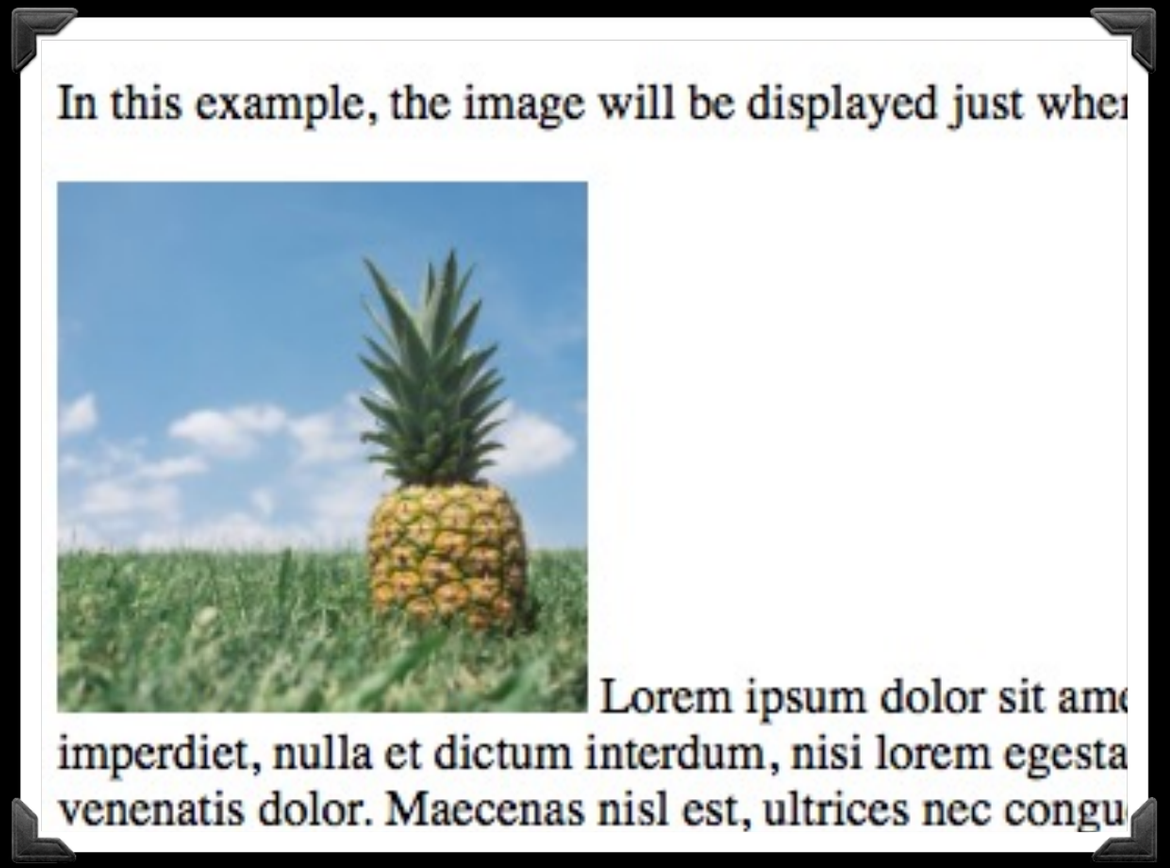
float:left;




```
<style>
img {
      float: left;
}
</style>
<p>
In this example, the image will float to the
   right in the paragraph, and the text in the
   paragraph will wrap around the image.</p>
<p><img src="pineapple.jpg"
   style="width:170px;height:170px;margin-
   right:15px;">
Lorem ipsum dolor sit amet, consectetur
   adipiscing elit……
```

float:none;

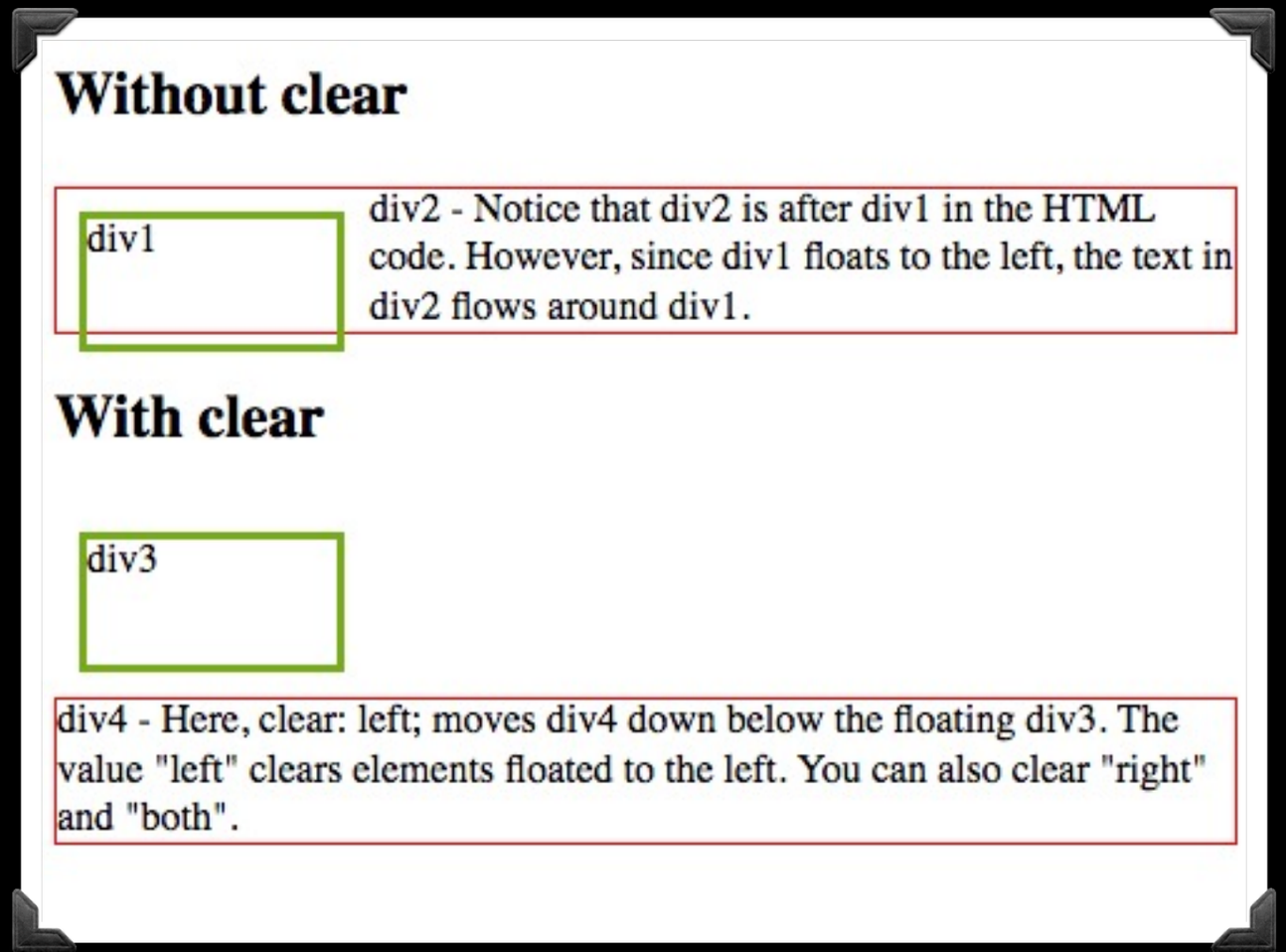
```
<style>
img {
   float: none;
}
</style>
<p>
In this example, the image will float to the
   right in the paragraph, and the text in the
   paragraph will wrap around the image.</p>
<p><img src="pineapple.jpg"
   style=“width:170px;height:170px;”>
Lorem ipsum dolor sit amet, consectetur
   adipiscing elit……
```



CSS 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.
- The clear property can have one of the following values:
  - ✓ None - Allows floating elements on both sides. This is default
  - ✓ 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
  - ✓ Inherit - The element inherits the clear value of its parent
- The most common way to use the clear property is after you have used a float property on an element.

```
<style>
.div1 {
  float: left;
  width: 100px;height: 50px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.div2 {
  border: 1px solid red;
}
.div3 {
  float: left;
  width: 100px;height: 50px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.div4 {
  border: 1px solid red;
  clear: left;
}
</style>
```



```
<h2>Without clear</h2>
<div class="div1">div1</div>
<div class="div2">div2 - Notice that div2
     is after div1 in the HTML code.
     However, since div1 floats to the left,
     the text in div2
   flows around div1.</div>
<h2>With clear</h2>
<div class="div3">div3</div>
<div class="div4">div4 - Here, clear: left;
     moves div4 down below the floating
     div3. The value "left" clears elements
     floated to the left. You can also clear
     "right" and "both".</div>
</body>
```

clearfix

```
<style>
div {
  border: 3px solid #4CAF50;
  padding: 5px;
}
.img1 {
  float: right;
}
.clearfix {
  overflow: auto;
}
.img2 {
  float: right;
}
</style>
```

In this example, the image is taller than the element containing it, and it is floated, so it overflows outside of its container:

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

Add a clearfix class with overflow: auto; to the containing element, to fix this problem:

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

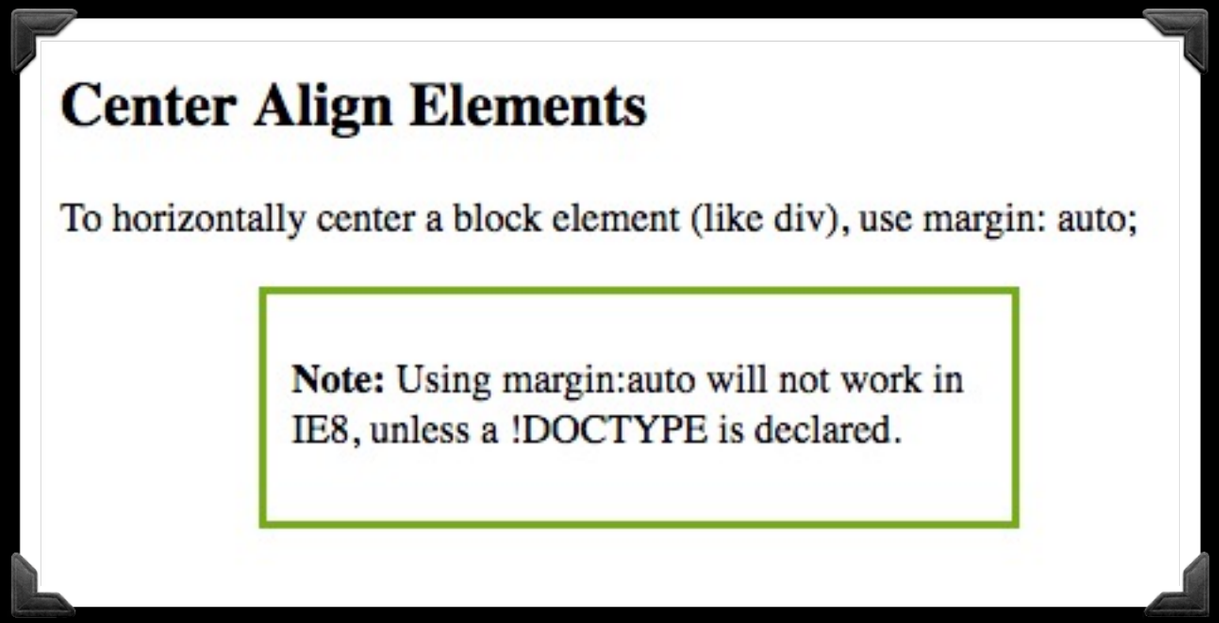
```
<p>In this example, the image is taller than the
     element containing it, and it is floated, so
     it overflows outside of its container:</p>
<div>
<img class="img1" src="pineapple.jpg"
     alt="Pineapple" width="170" height="170">
Lorem ipsum dolor sit amet, consectetur
     adipiscing elit. Phasellus imperdiet, nulla
     et dictum interdum...</div>
```

```
<p style="clear:right">Add a clearfix class with
     overflow: auto; to the containing element,
     to fix this problem:</p>

<div class="clearfix">
<img class="img2" src="pineapple.jpg"
     alt="Pineapple" width="170" height="170">
Lorem ipsum...</div>
```

CSS Layout - Horizontal & Vertical Align

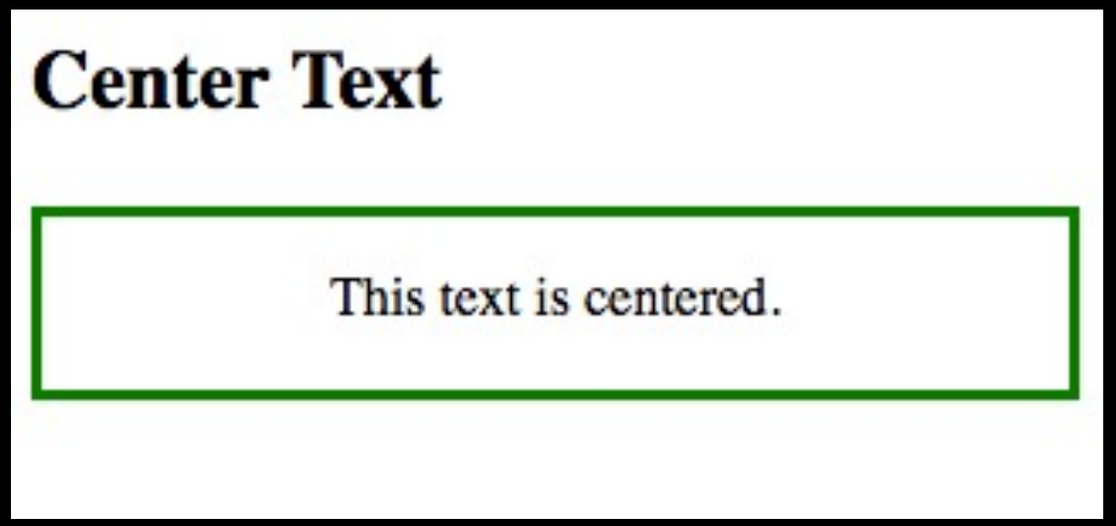
Center Align Elements

```
<style>
.center {
  margin: auto;
  width: 60%;
  border: 3px solid green;
  padding: 10px;
}
</style>
<h2>Center Align Elements</h2>
<p>To horizontally center a block element (like
    div), use margin: auto;</p>
<div class="center">
<p><b>Note: </b>Using margin:auto will not work
        in IE8, unless a !DOCTYPE is declared.
</p>
</div>
```



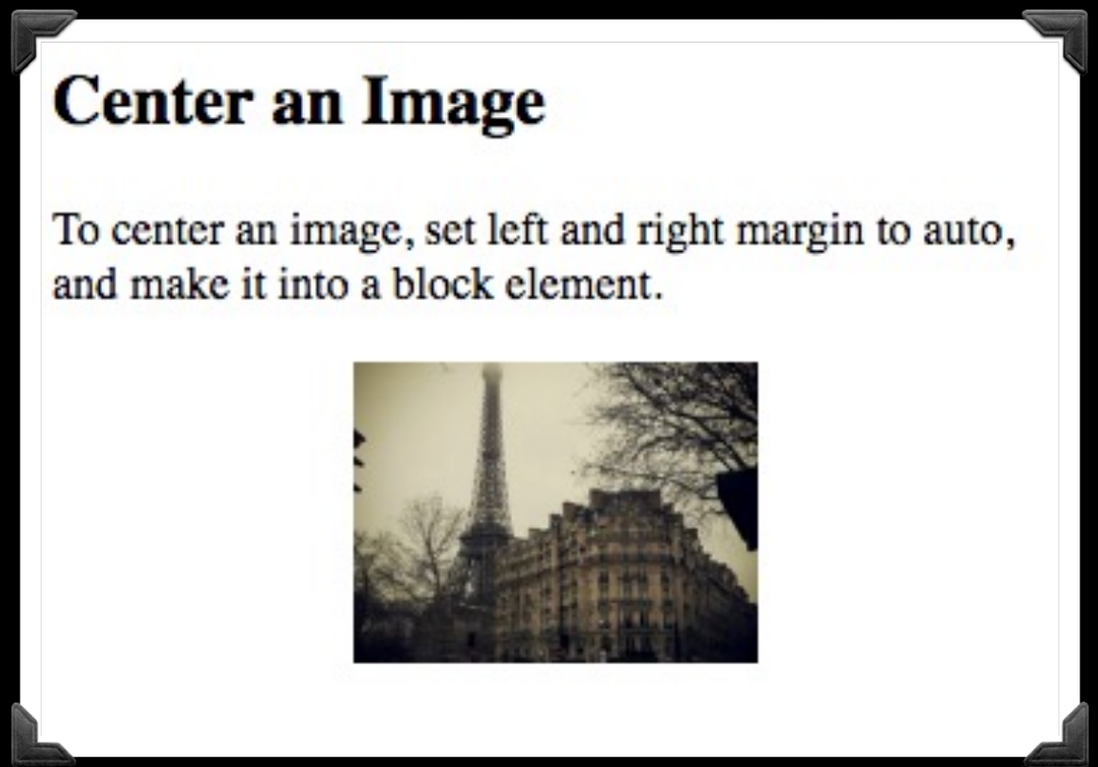
Center Align Text

```
<style>
.center {
text-align: center;
border: 3px solid green;
}
</style>
<h2>Center Text</h2>
<div class="center">
<p>This text is centered.</p>
</div>
```



Center an Image

```
<style>
img {
display: block;
margin-left: auto;
margin-right: auto;
}
</style>
```




```
<h2>Center an Image</h2>
<p>To center an image, set left and right
    margin to auto, and make it into a
    block element.</p>
<img src="paris.jpg" alt="Paris"
    style="width:40%">
```

CSS Pseudo-classes

CSS Pseudo-classes

What are Pseudo-classes?

A pseudo-class is used to define a special state of an element.

For example, it can be used to:

- ✓ Style an element when a user mouses over it
- ✓ Style visited and unvisited links differently
- ✓ Style an element when it gets focus

The syntax of pseudo-classes:

```
selector:pseudo-class{
        property:value;
        }
```

Anchor Pseudo-classes

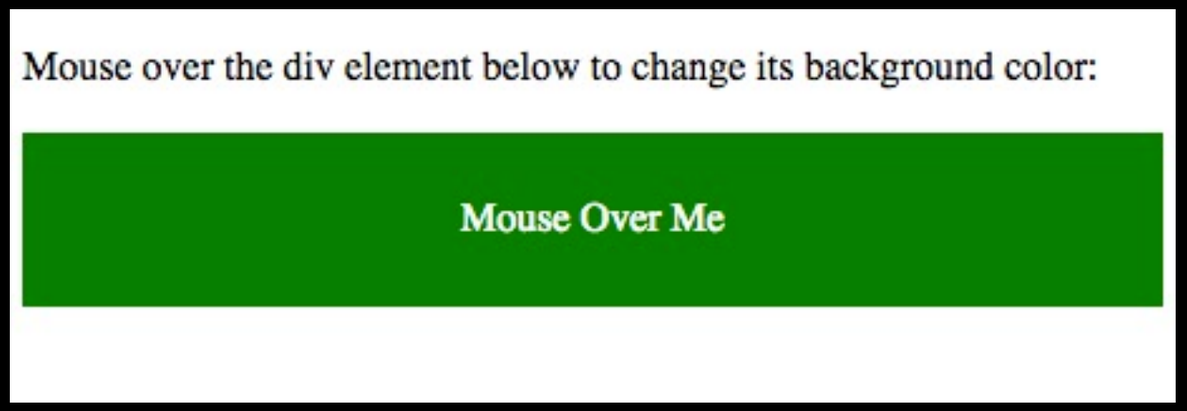
```
<style>
/* unvisited link */
a:link {
 color: red;
}
/* visited link */
a:visited {
 color: green;
}
/* mouse over link */
a:hover {
 color: yellow;
}
/* selected link */
a:active {
 color: blue;
}
</style>
```

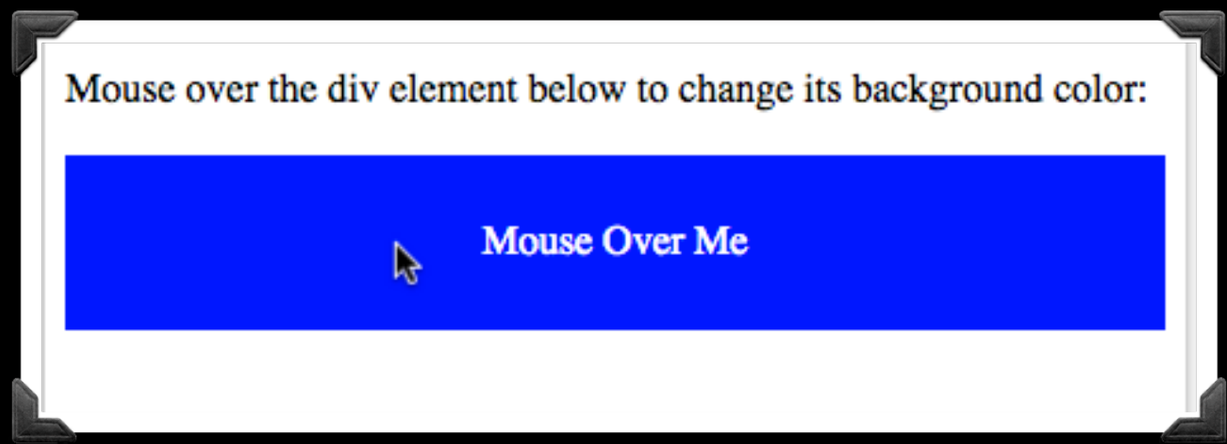
Pseudo-classes and CSS Classes

```
<style>
a.highlight:hover {
        color: #ff0000;
}
</style>
<body>
<p>
<a class="highlight"
    href="css_syntax.asp">CSS Syntax</a></
    p>
<p>
<a href="default.asp">CSS Tutorial</a></p>
</body>
```

Hover on

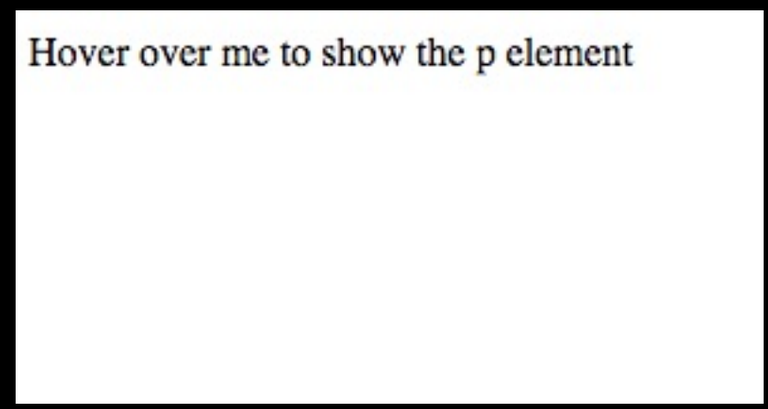
```
<style>
div {
background-color:green;
color: white;
padding: 25px;
text-align: center;
}
div:hover {
background-color:blue;
}
</style>
<p>Mouse over the div element below to
    change its background color:</p>
<div>Mouse Over Me</div>
```

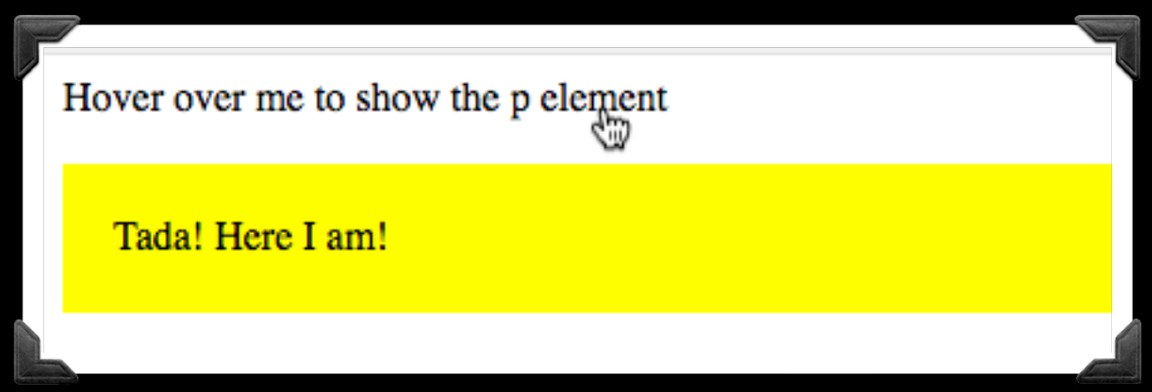




Simple Tooltip Hover

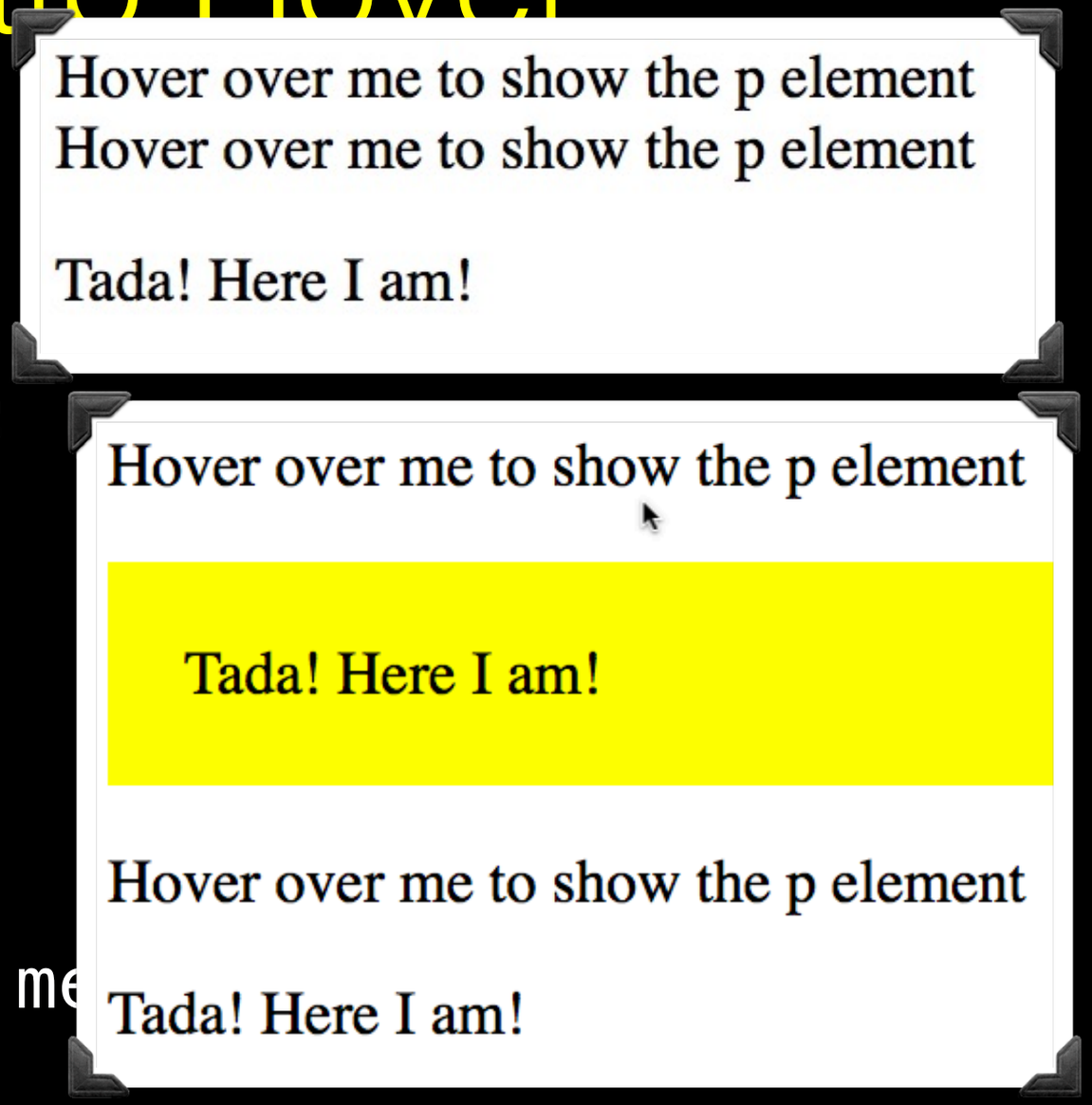
```
<style>
p {
 display: none;
 background-color: yellow;
 padding: 20px;
}
div:hover p {
 display: block;
}
</style>
<div>Hover over me to show the p element
<p>Tada! Here I am!</p>
</div>
```





Simple Tooltip Hover

```
<style>
.p1 {
    display: none;
    background-color: yellow;
    padding: 20px;
}
.div1:hover .p1{
    display: block;
}
</style>
<div class="div1">Hover over me
    element
  <p class="p1">Tada! Here I am!</p>
</div>
<div>Hover over me to show the p element
  <p>Tada! Here I am!</p>
</div>
```



CSS - The :first-child Pseudo-class
Match the first <p> element

```
<style>
p:first-child {
        color: blue;
}
</style>
</head>
<body>
<p>This is some text.</p>
<p>This is some text.</p>
<p><b>Note:</b> For :first-child to work in
    IE8 and earlier, a DOCTYPE must be
    declared.</p>
```

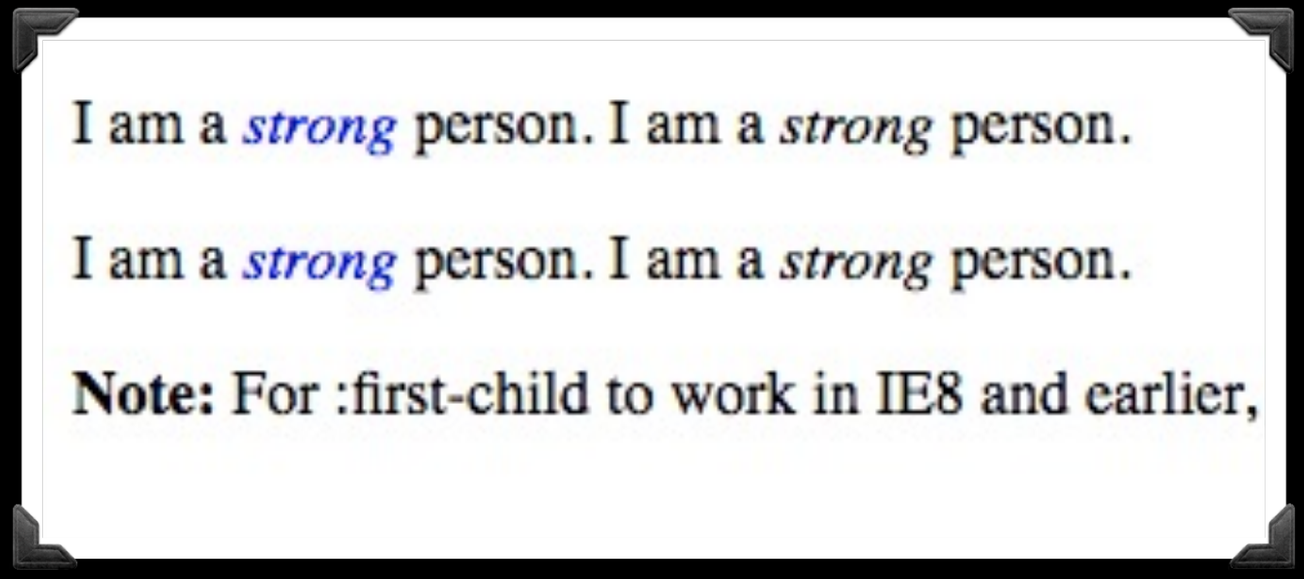
This is some text.

This is some text.

Note: For :first-child to work in IE8 and earlier, a DOCTYPE must be declared.

Match the first <i> element in all <p> elements

```
<style>
p i:first-child {
        color: blue;
}
</style>
<p>I am a <i>strong</i> person. I am a
    <i>strong</i> person.</p>

<p>I am a <i>strong</i> person. I am a
    <i>strong</i> person.</p>

<p><b>Note:</b> For :first-child to work in
    IE8 and earlier, a DOCTYPE must be
    declared.</p>
```



CSS Navigation Bar

CSS Navigation Bar

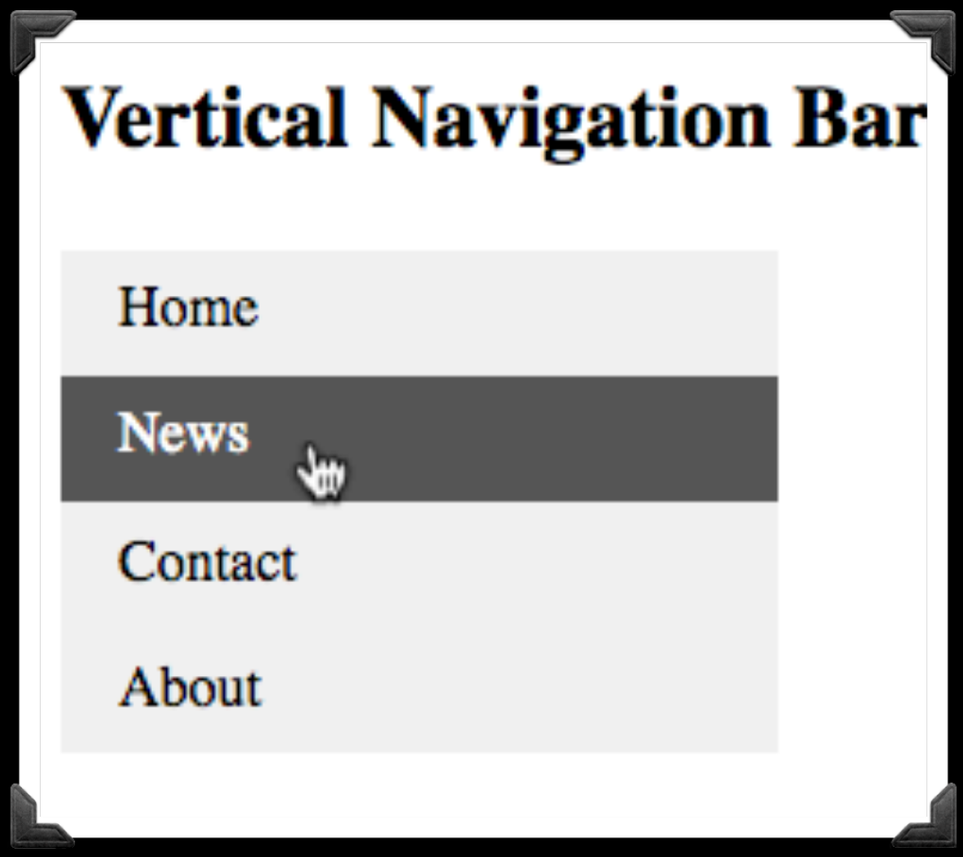
- Having easy-to-use navigation is important for any web site.
- With CSS you can transform boring HTML menus into good-looking navigation bars.

- Navigation Bar = List of Links
  - ✓ A navigation bar needs standard HTML as a base.

CSS Navigation Bar - Vertical Navigation Bar

Vertical Navigation Bar

```
<style>
ul {
list-style-type: none;
margin: 0;  padding: 0;
width: 200px;
background-color: #f1f1f1;
}
li a {
display: block;
color: #000;
padding: 8px 16px;
text-decoration: none;
}
/* Change the link color on hover */
li a:hover {
background-color: #555;
color: white;
}
</style>
```

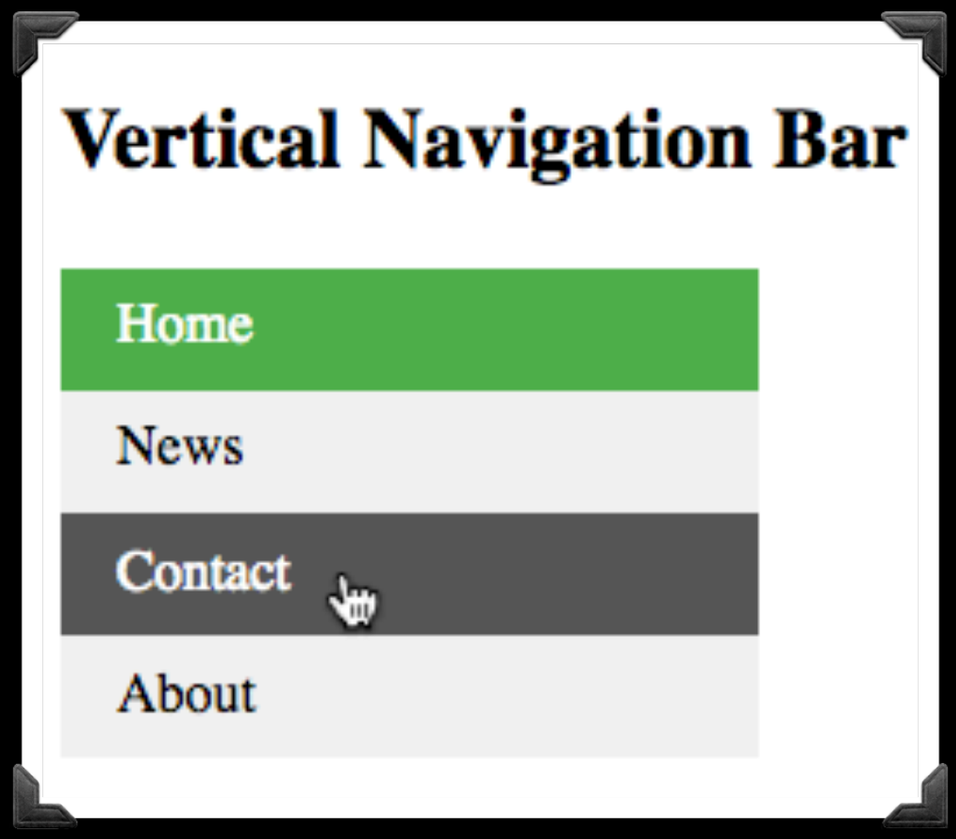


```
<h2>Vertical Navigation Bar</h2>
<ul>
<li><a href="#home">Home</a></li>
<li><a href="#news">News</a></li>
<li><a href="#contact">Contact</a></li>
<li><a href="#about">About</a></li>
</ul>
```

```
<style>
ul {
list-style-type: none;
margin: 0;  padding: 0;
width: 200px;
background-color: #f1f1f1;
}
li a {
display: block;
color: #000;
padding: 8px 16px;
text-decoration: none;
}
li a.active {
background-color: #4CAF50;
color: white;
}
li a:hover:not(.active) {
background-color: #555;
color: white;
}
</style>
```

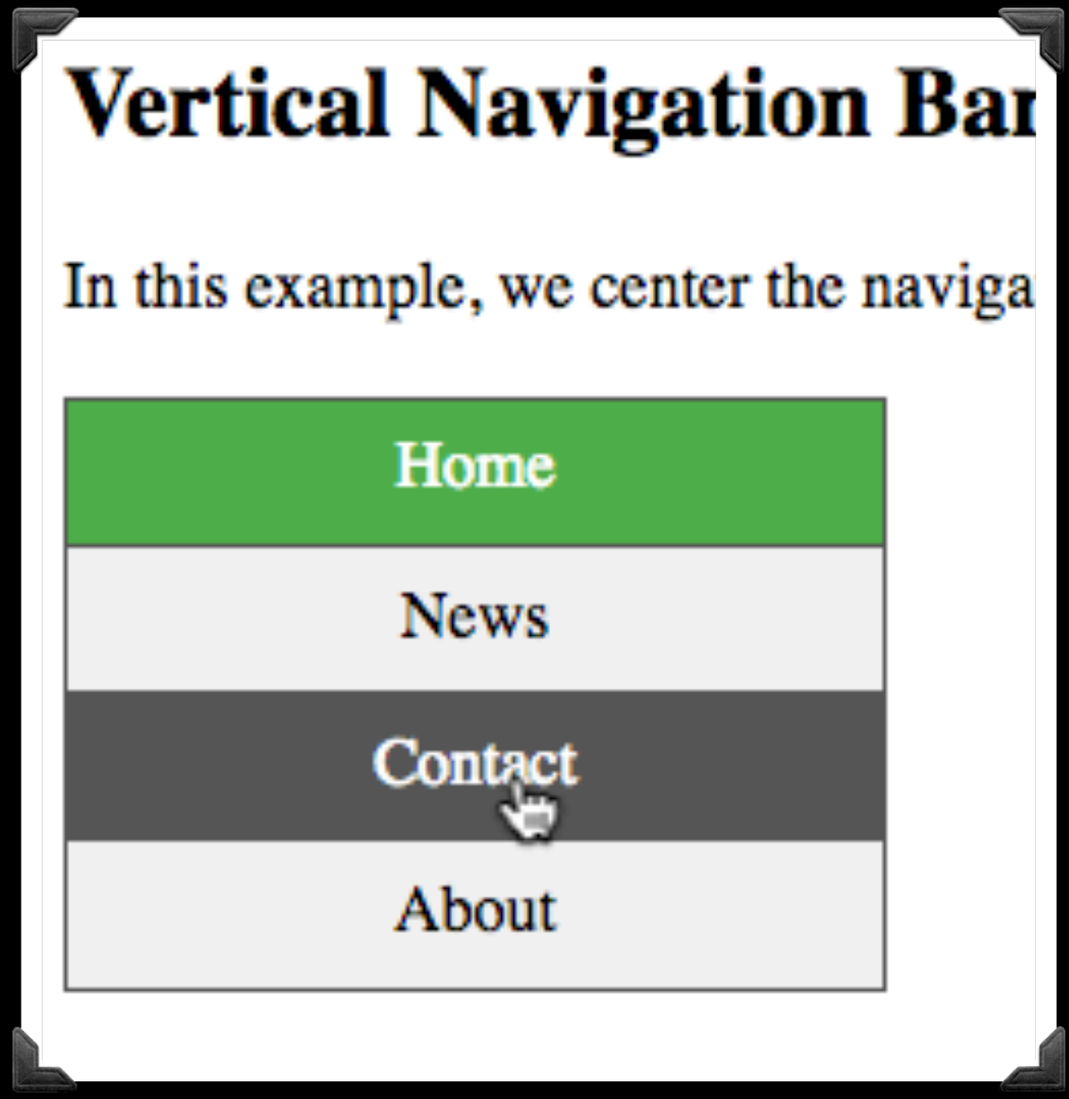
Active/Current Navigation Link

In this example, we create an "active" class with a green background color and a white text. The class is added to the "Home" link.



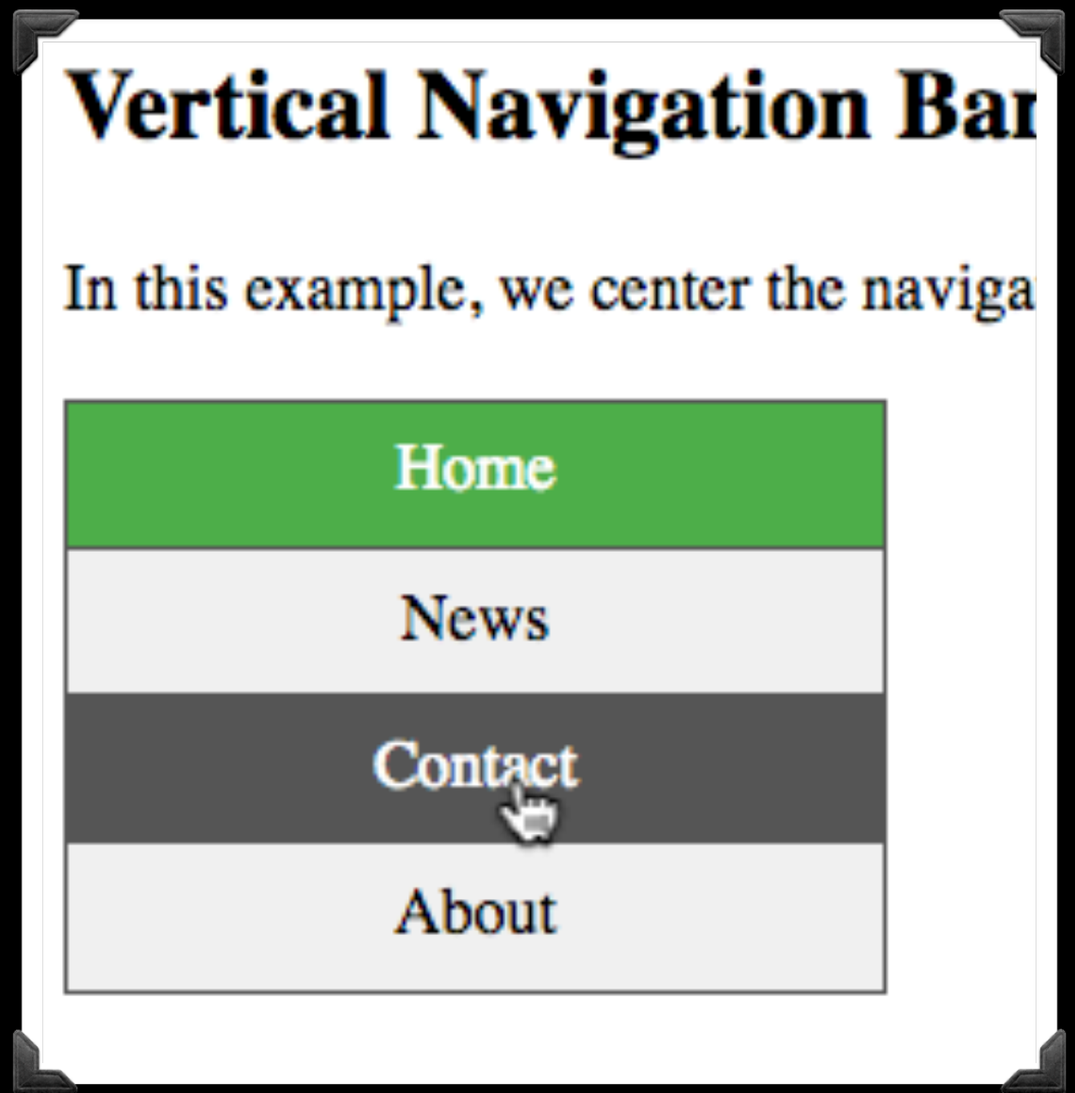
Center Links & Add Borders

```
<style>
ul {
list-style-type: none;
margin: 0;
padding: 0;
width: 200px;
background-color: #f1f1f1;
border: 1px solid #555;
}

li a {
display: block;
color: #000;
padding: 8px 16px;
text-decoration: none;
}
```



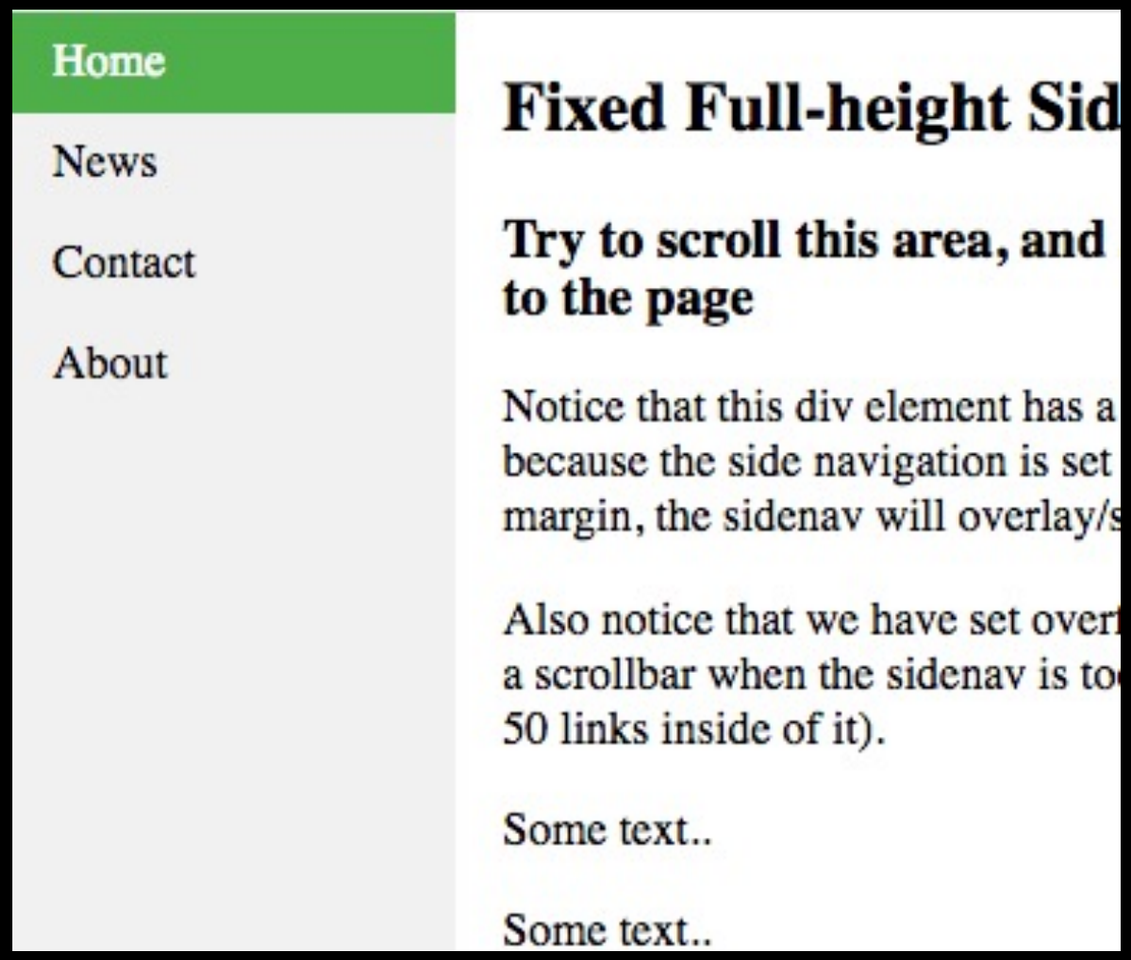
Center Links & Add Borders

```
li {
text-align: center;
border-bottom: 1px solid #555;
}

li:last-child {
border-bottom: none;
}

li a.active {
background-color: #4CAF50;
color: white;
}

li a:hover:not(.active) {
background-color: #555;
color: white;
}
</style>
```



Fixed Full-height Side Nav

```
<style>
body {
margin: 0;
}
ul {
list-style-type: none;
margin: 0; padding: 0;
width: 25%; height: 100%;
background-color: #f1f1f1;
position: fixed; overflow: auto;
}
li a {
display: block;
color: #000;
padding: 8px 16px;
text-decoration: none;
}

li a.active {
background-color: #4CAF50;
color: white;
}
```



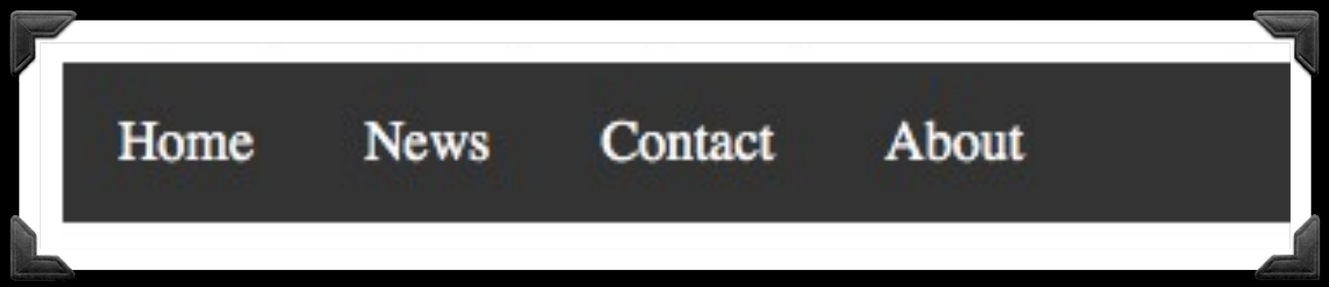
```
li a:hover:not(.active) {
background-color: #555;
color: white;
}
</style>
<ul>
<li><a class="active" href="#home">Home</
    a></li>
<li><a href="#news">News</a></li>
<li><a href="#contact">Contact</a></li>
<li><a href="#about">About</a></li>
</ul>
<div style="margin-left:25%;padding:1px
    16px;height:1000px;">
<h2>Fixed Full-height Side Nav</h2>
<h3>Try to scroll this area, and see how the
    sidenav sticks to the page</h3>
```

CSS Navigation Bar - Horizontal Navigation Bar

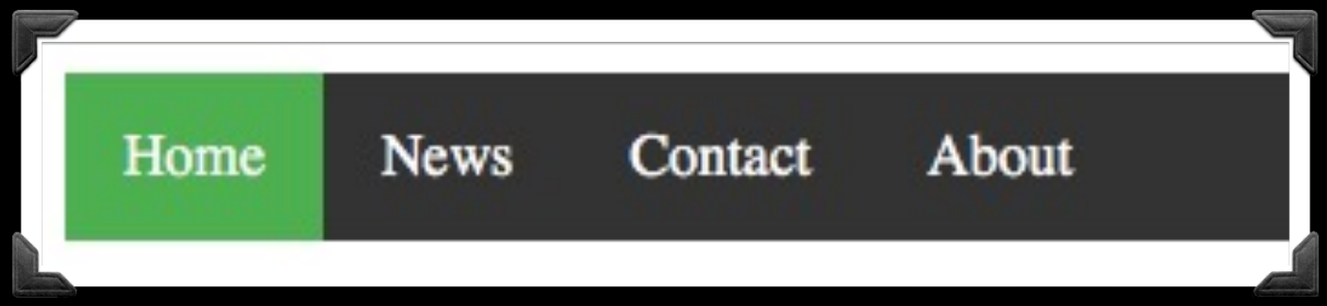
Horizontal Navigation Bar

```
<style>
ul {
    list-style-type: none;
    margin: 0;  padding: 0;
    overflow: hidden;
}
li {
    float: left;  }
li a {
    display: block;
    padding: 8px;
    background-color: #dddddd;
}
</style>
<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>
```

```
<style>
ul {
    list-style-type: none;
    margin: 0;  padding: 0;
    overflow: hidden;
    background-color: #333333;
}
li {
    float: left; }
li a {
    display: block; color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover {
    background-color: #111; }
</style>
<ul> <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="“#about">About</a></li> </ul>
```

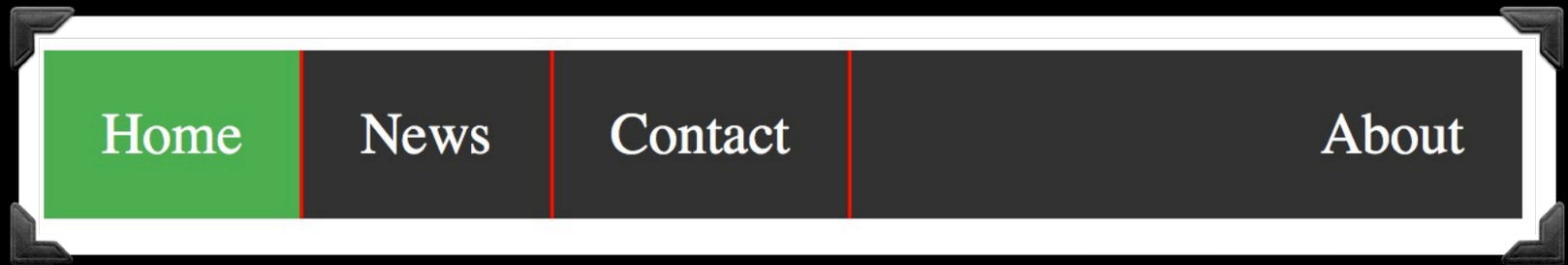


```
<style>
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333; }
li {
    float: left;  }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul>  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href=“#about">About</a></li> </ul>
```

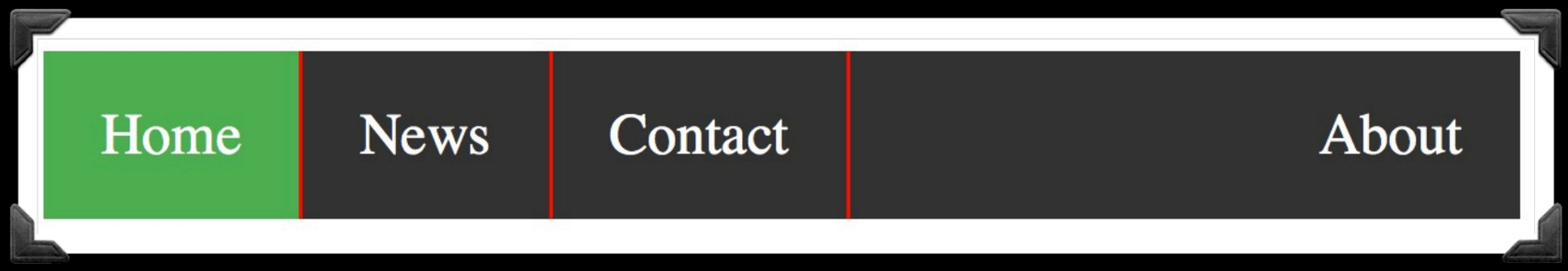


```
<style>
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333; }
li {
    float: left; }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul>  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li style="float:right"><a class="active" href="#about">About</
      a></li> </ul>
```

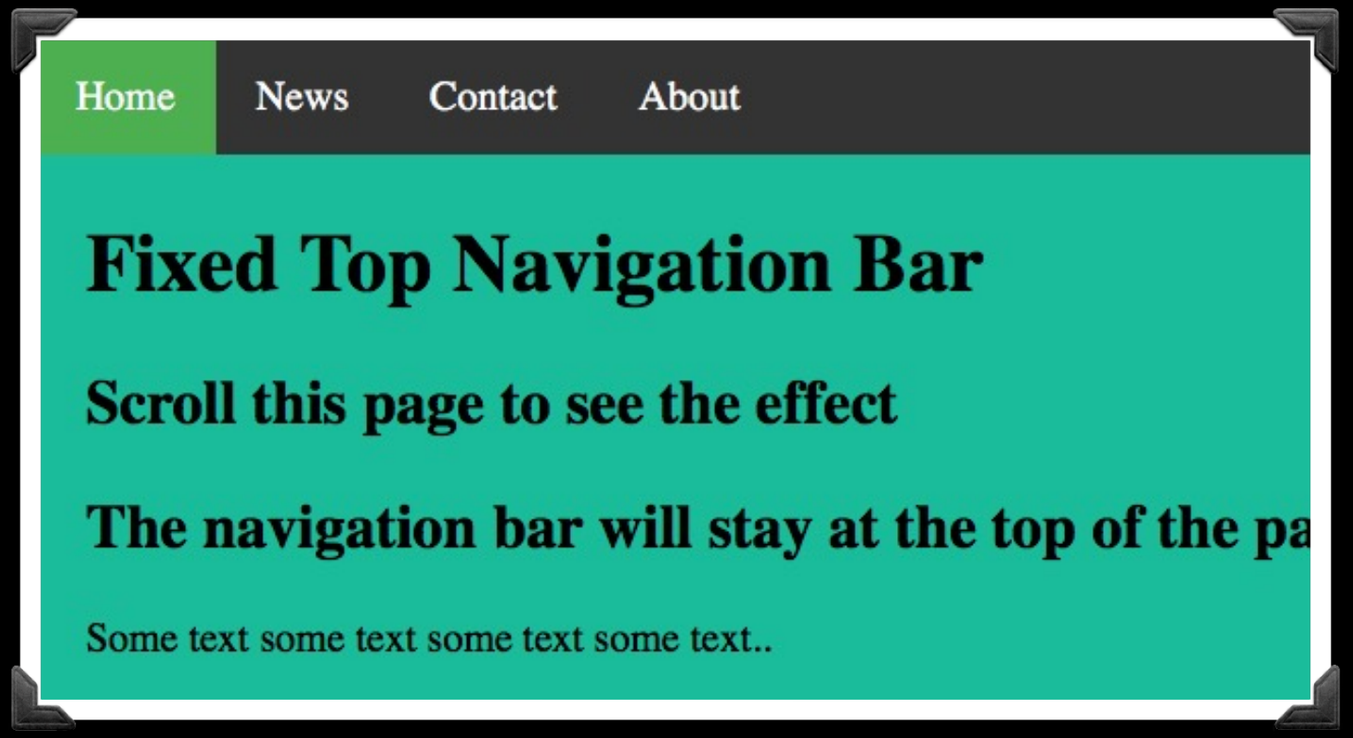
```
<style>
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333; }
li {
    float: left;
    border-right:1px solid red; }
li:last-child {
    border-right: none; }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul> <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li style="float:right"><a href="#about">About</a></li> </ul>
```



```
<style>
body {margin:0;}
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: auto;
    background-color: #333;
    position: fixed;
    top: 0;
    width: 100%; }
li {
  float: left; }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul>  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="“#about">About</a></li> </ul>
<div style="padding:20px;margin-top:30px;background-
    color:#1abc9c;height:1500px;">
<h1>Fixed Top Navigation Bar</h1>
```



```
<style>
body {margin:0;}
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333;
    position: fixed;
    top: 0;
    width: 100%; }
li {
    float: left; }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul>  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="“#about">About</a></li> </ul>
<div style="padding:20px;background-color:#1abc9c;height:1500px;">
<h1>Fixed Bottom Navigation Bar</h1>
```



```
<style>
body {margin:0;}
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333;
    position: fixed;
    bottom: 0;
    width: 100%; }
li {
    float: left; }
li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none; }
li a:hover:not(.active) {
    background-color: #111; }
.active {
    background-color: #4CAF50; }
</style>
<ul>  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="“#about">About</a></li> </ul>
<div style="padding:20px;background-color:#1abc9c;height:1500px;">
<h1>Fixed Bottom Navigation Bar</h1>
```

CSS Image Sprites

- An image sprite is a collection of images put into a single image.
- A web page with many images can take a long time to load and generates multiple server requests.
- Using image sprites will reduce the number of server requests and save bandwidth.

CSS Image Sprites

```
<style>
#home {
width: 46px;
height: 44px;
background: url(img_navsprites.gif) 0 0;
}
#next {
width: 43px;
height: 44px;
background: url(img_navsprites.gif) -91px 0;
}
</style><body>
<img id="home" src="img_trans.gif"><br><br>
<img id="next" src=“img_trans.gif”> </body>
```

- Instead of using three separate images, we use this single image ("img_navsprites.gif")
- With CSS, we can show just the part of the image we need.
- In the following example the CSS specifies which part of the "img_navsprites.gif" image to show
  - ✓ <img id="home" src="img_trans.gif"> - Only defines a small transparent image because the src attribute cannot be empty. The displayed image will be the background image we specify in CSS
  - ✓ width: 46px; height: 44px; - Defines the portion of the image we want to use
  - ✓ background: url(img_navsprites.gif) 0 0; - Defines the background image and its position (left 0px, top 0px)

CSS Attribute Selectors

CSS Attribute Selectors

- Style HTML Elements With Specific Attributes
- It is possible to style HTML elements that have specific attributes or attribute values.

CSS [attribute] Selector

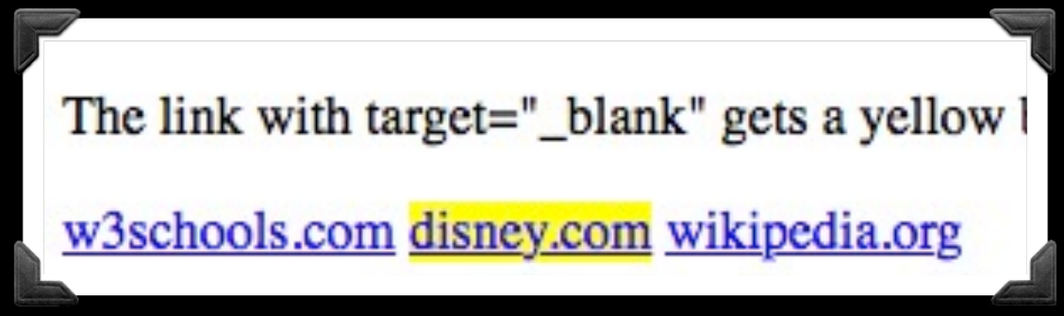
The [attribute] selector is used to select elements with a specified attribute.


```
<style>
a[target]
{
      background-color: yellow;
      }
</style>
<p>The links with a target attribute gets a
      yellow background:</p>
<a href="https://
      www.w3schools.com">w3schools.com</a>
<a href="http://www.disney.com"
      target="_blank">disney.com</a>
<a href="http://www.wikipedia.org"
      target="_top">wikipedia.org</a>
```

CSS [attribute="value"] Selector

The [attribute="value"] selector is used to select elements with a specified attribute and value.

```
<style>
a[target=_blank] {
    background-color: yellow;
}
</style>
```



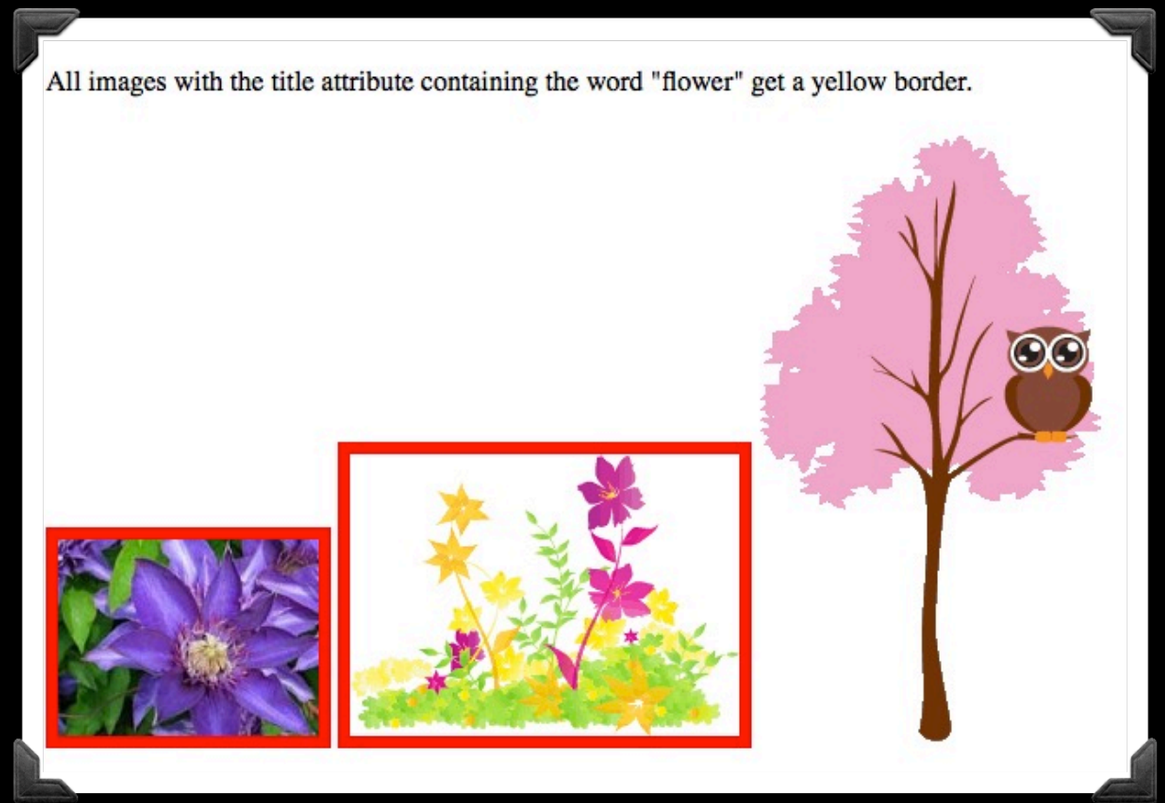

```
<p>The link with target="_blank" gets a yellow
    background:</p>
<a href="https://
    www.w3schools.com">w3schools.com</a>
<a href="http://www.disney.com"
    target="_blank">disney.com</a>
<a href="http://www.wikipedia.org"
    target="_top">wikipedia.org</a>
```

CSS [attribute~="value"] Selector

- The [attribute~="value"] selector is used to select elements with an attribute value containing a specified word.
- The following selects all elements with a title attribute that contains a space-separated list of words, one of which is “flower”.
- It will match elements with title="flower", title="summer flower", and title="flower new", but not title="my-flower" or title=“flowers"

CSS [attribute~="value"] Selector

```
<style>
[title~=flower] {
border: 7px solid red;
}
</style>
</head>
<body>
<p>All images with the title attribute containing
      the word "flower" get a yellow border.</p>
<img src="klematis.jpg" title="klematis flower"
      width="150" height="113">
<img src="img_flwr.gif" title="flower" width="224"
      height="162">
<img src="img_tree.gif" title="tree" width="200"
      height="358">
```



CSS [attribute|="value"] Selector

- The **[attribute|="value"]** selector is used to select elements with the specified attribute starting with the specified value.
- The following selects all elements with a class attribute value that begins with “top".
- Note: The value has to be a whole word, either alone, like class="top", or followed by a hyphen(-), like class="top-text"!

CSS [attribute|="value"] Selector

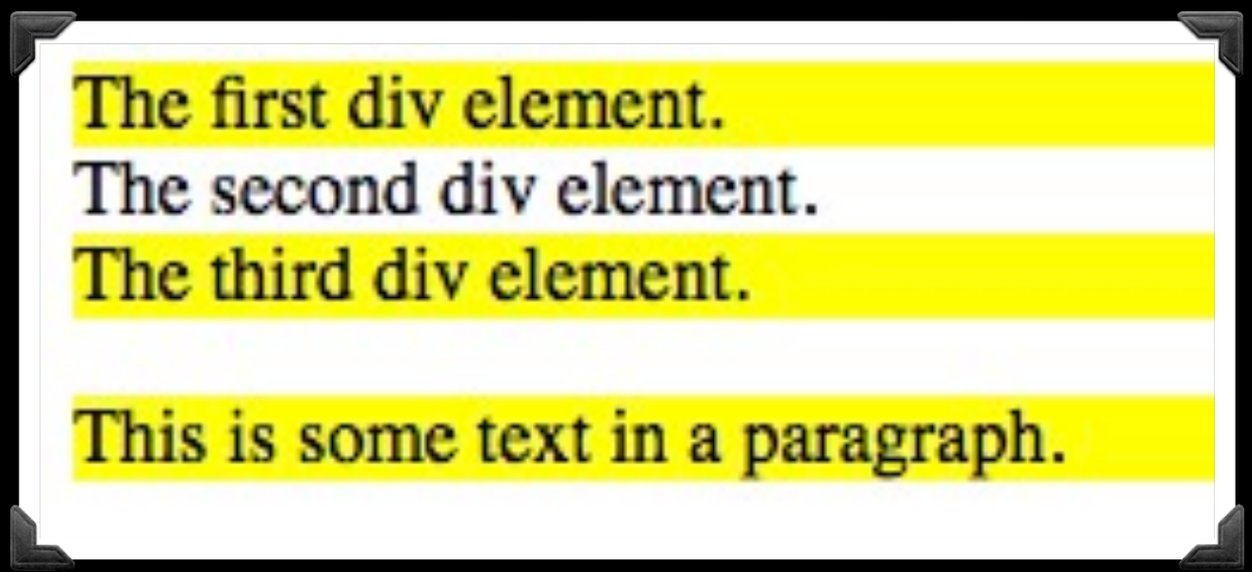
```
<style>
[class|=top] {
  background-color: yellow;
}
</style>
<h1 class="top-header">Welcome</h1>
<p class="top-text">Hello world!</p>
<p class=“topcontent">Are you learning
    CSS?
</p>
```

CSS [attribute$="value"] Selector

- The **[attribute$="value"]** selector is used to select elements whose attribute value ends with a specified value.
- The following selects all elements with a class attribute value that ends with “test”.
- Note: The value does not have to be a whole word!

CSS [attribute$="value"] Selector

```
<style>
[class$="test"] {
        background: yellow;
}
</style>
<div class="first_test">The first div element.</div>
<div class="second">The second div element.</div>
<div class="my-test">The third div element.</div>
<p class="mytest">This is some text in a
      paragraph.</p>
```

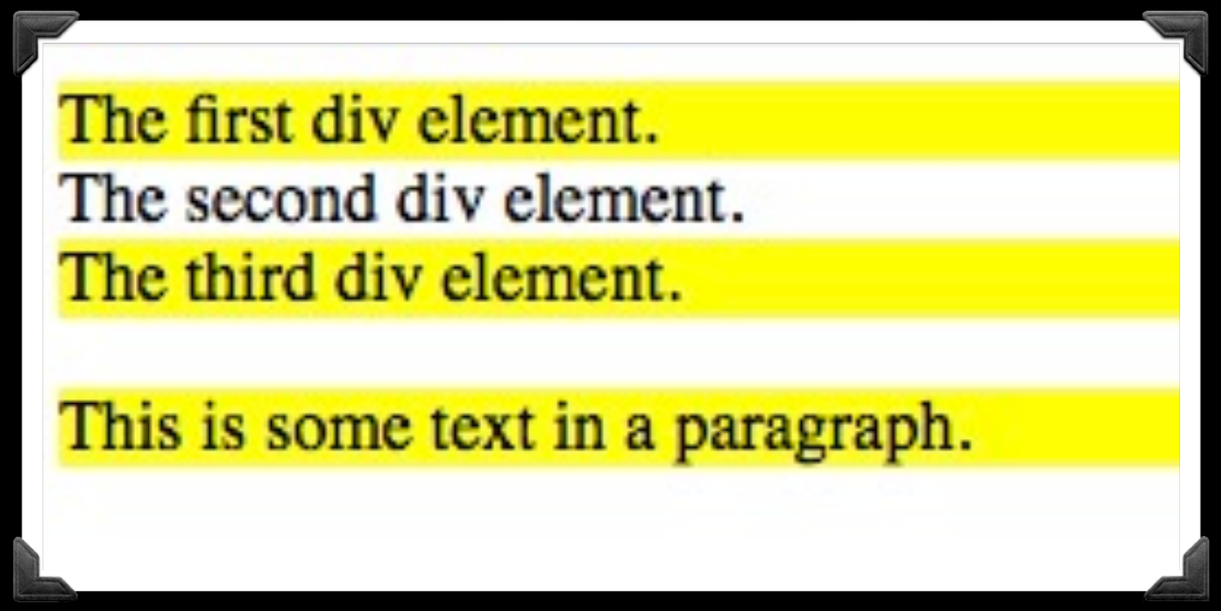


CSS [attribute*="value"] Selector

- The [attribute*="value"] selector is used to select elements whose attribute value contains a specified value.
- The following selects all elements with a class attribute value that contains “te".
- Note: The value does not have to be a whole word!

CSS [attribute*="value"] Selector

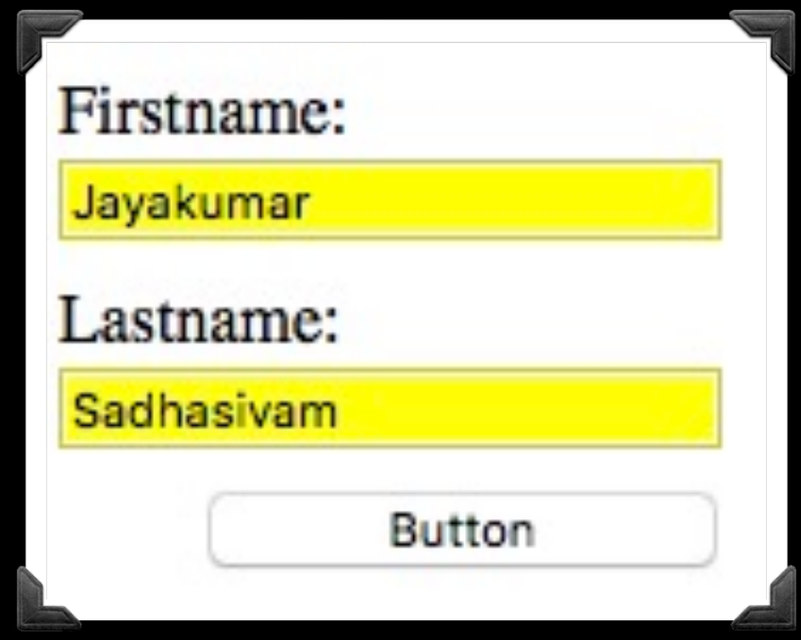
```
<style>
[class*="te"] {
        background: yellow;
}
</style>
<div class="first_test">The first div element.</div>
<div class="second">The second div element.</div>
<div class="my-test">The third div element.</div>
<p class="mytestsss">This is some text in a
      paragraph.</p>
```



Styling Forms

The attribute selectors can be useful for styling forms without class or ID:

```
<style>
input[type=text] {
width: 150px; display: block;
margin-bottom: 10px;
background-color: yellow;
}
input[type=button] {
width: 120px; margin-left: 35px;
display: block;
}
</style>
<form name="input" action="" method="get">
Firstname:<input type="text" name="Name" value="Jayakumar">
Lastname:<input type="text" name="Name" value="Sadhasivam">
<input type="button" value="Example Button”> </form>
```



Creating Page Layout and Design

Steps

1. Header
2. Menu Bar
3. Adding 3 Columns to display content
4. First and Last Column for Side Bar and Middle column for main page
5. Footer

Header

Resize the browser window to see the responsive effect.

Link Link Link

Side

Lorem ipsum dolor sit amet, consectetur adipiscing elit..

Main Content

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas sit amet pretium urna. Vivamus venenatis velit nec neque ultricies, eget elementum magna tristique. Quisque vehicula, risus eget aliquam placerat, purus leo tincidunt eros, eget luctus quam orci in velit. Praesent scelerisque tortor sed accumsan convallis.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas sit amet pretium urna. Vivamus venenatis velit nec neque ultricies, eget elementum magna tristique. Quisque vehicula, risus eget aliquam placerat, purus leo tincidunt eros, eget luctus quam orci in velit. Praesent scelerisque tortor sed accumsan convallis.

Side

Lorem ipsum dolor sit amet, consectetur adipiscing elit..

Footer

Reference

- https://www.w3schools.com/css/
- https://www.tutorialspoint.com/css/