Front-end <Web /> Development

Lesson 3: CSS (Part 1)

University of Tehran

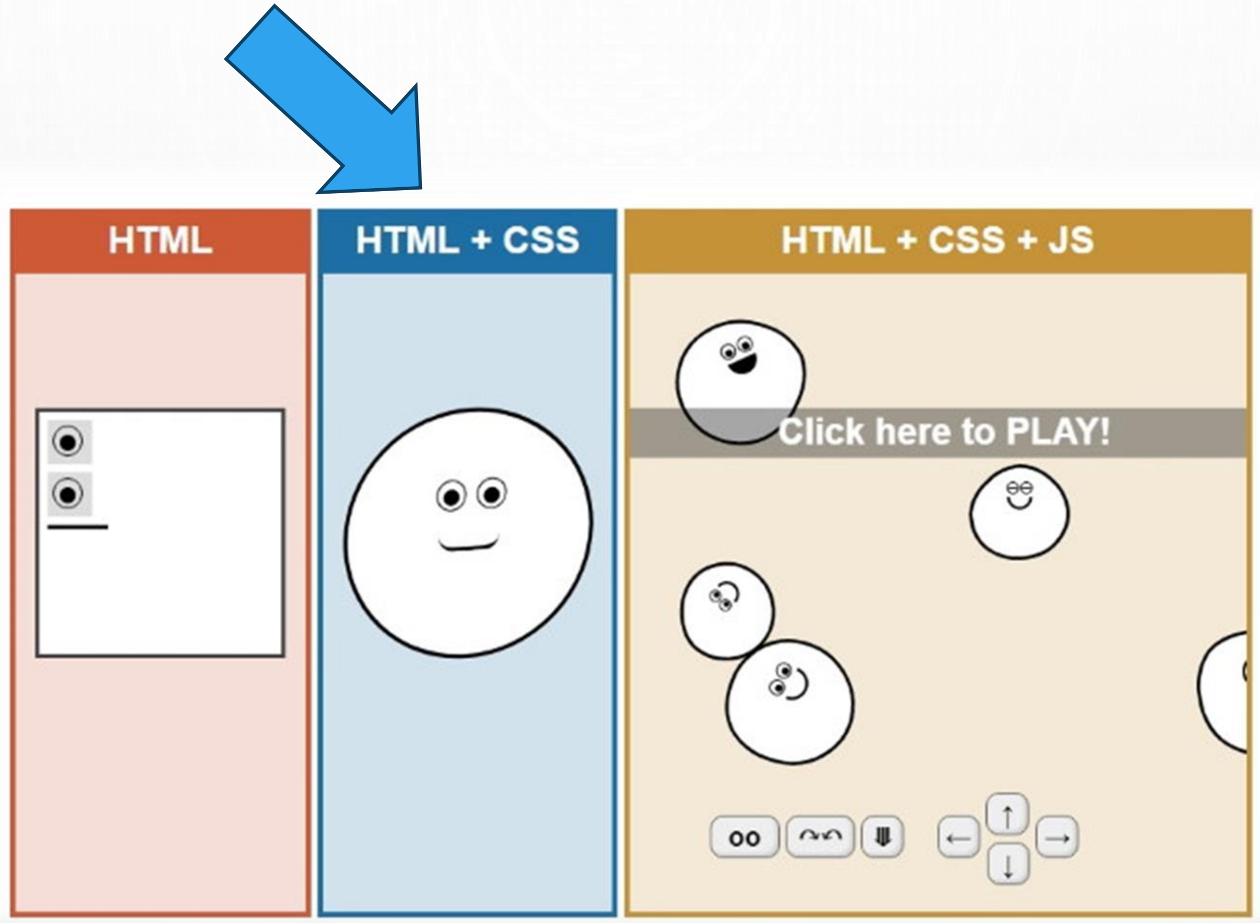
ACM
Summer of Code 2024

Lecturer:
Misagh Mohaghegh

A Short Review…

- Hypertext Markup Language
- Elements, opening and closing tags, content
- Nested elements, void elements
- Attributes, global attributes
- Comments, XHTML
- Page anatomy (html, head, body)
- Metadata
- Body
- Semantic tags
- Forms and inputs

Time for…



Cascading Style Sheets

CSS & Selectors

Cascading Style Sheets

The primary tool to style web pages.

- **Cascading:** Multiple styles can be applied to an element. Cascading rules choose which one overrides the others.
- **Style Sheets:** A sheet that describes the styles of elements.

Version History: (W3C recommendation)

- **CSS 1** (1996): Fonts, colors, spacing, box model.
- **CSS 2** (1998): Positioning, media types, more properties.
- **CSS 2.1** (2011): Fixing most problems, add browser specific extensions.
- **CSS 3** (2012 - ...): Features are divided and modularized. Each module is standardized separately. This allows for faster development and browser implementation. Modules include Selectors, Backgrounds, Box Model, Color, Fonts, Flexbox, Grid Layout, Animations, Transformations and more.

Adding CSS

CSS rules are mostly written in a **file.css** and linked in the corresponding **file.html**. (External CSS)

```
<head>
    <link rel="stylesheet" type="text/css" href="file.css">
</head>
```

We can also write the CSS directly in the HTML file: (Internal CSS)

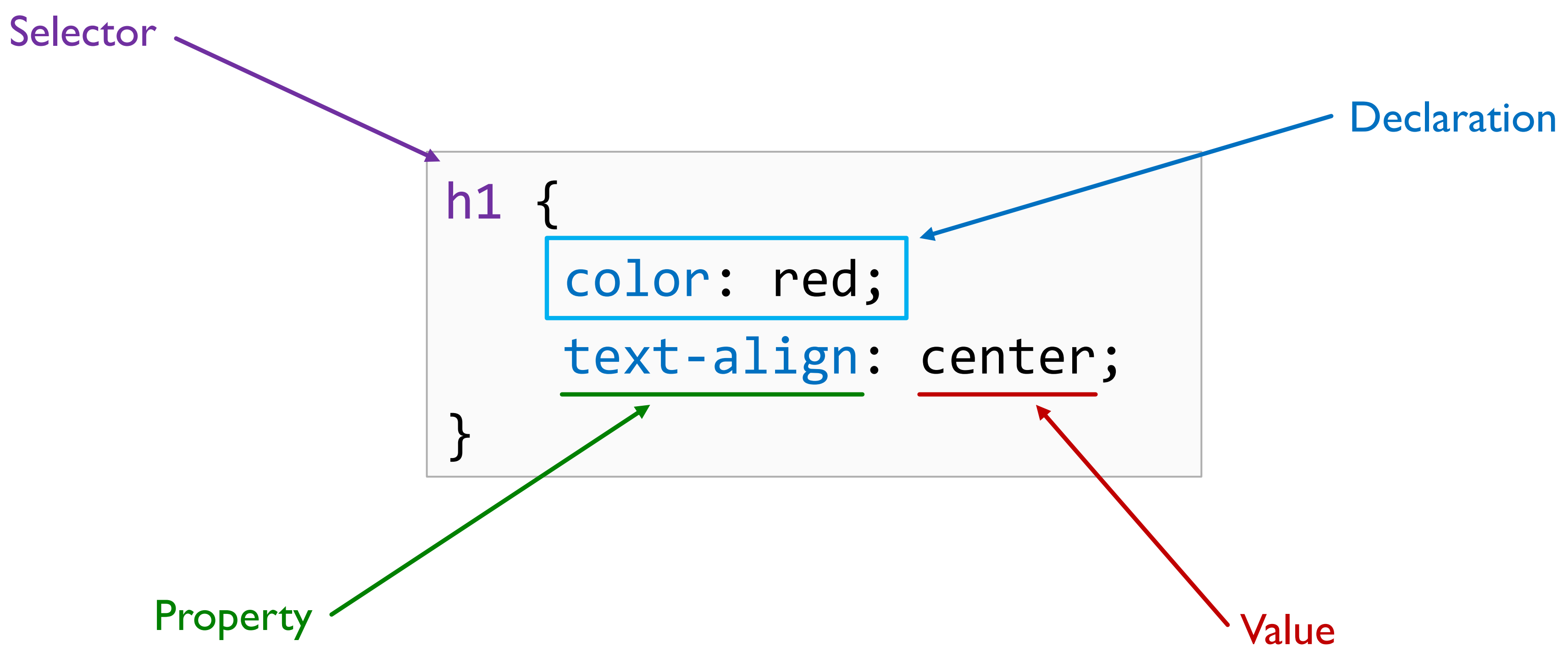
```
<head>
    <style>
        /* CSS written here */
    </style>
</head>
```

Or apply CSS directly to an element: (Inline CSS)

```
<p style="/* CSS written here */">Test</p>
```

CSS Rules

A CSS rule has the following structure:



```
h1 {
    color: red;
    text-align: center;
}
```

Selectors

We have to first select an HTML element to apply styles to it.

There are many ways to select elements:

- Simple selectors
- Attribute selectors
- Combinator selectors
- Pseudo-class selectors
- Pseudo-element selectors

Recall comments in HTML code: `<!-- This is a comment! -->`

Comments in CSS code: `/* This is a comment! */`

Simple Selectors

- Tag selector:

```
h1 { ... }
p { ... }
ul { ... }
img { ... }
```

```
<h1>Test</h1>
```

- ID selector:

```
#id { ... }
#my-image { ... }
```

```
<anytag id="id">Test</anytag>
<!-- The element with #id is selected
(there is only one) -->
```

- Class selector:

```
.class { ... }
.bottom-text { ... }
```

```
<anytag class="class">Test</anytag>
<!-- All elements that have this class
are selected -->
```

Simple Selectors

- Universal selector:

```
* { ... }
```

```
<h1>Test</h1>
<p>All elements are selected</p>
```

- In combination:

```
p#title.center { ... }
```

```
<p id="title" class="center">Test</p>
```

- Grouping selector:

```
h1.main,
p {
    ...
}
```

```
<h1 class="main center">Test</h1>
<p>All p tags and h1 elements that
have the "main" class get the same
property values</p>
```

Attribute Selectors

• Have attribute:	`[type] { ... }`	`<input type="text">`
• In combination:	`input.rev[type] { ... }`	`<input class="rev" type="email">`
• Attribute value:	`[type="url"] { ... }`	`<input type="url">`
• Space separated:	`[title~="Web"] { ... }`	`<abbr title="World Wide Web">`
• Hyphen separated:	`[class|="top"] { ... }`	`<p class="top-panel">Text</p>`
• Case insensitive:	`[value="del" i] { ... }`	`<input type="button" value="DEL">`

Attribute Selectors

• Starts with:	`[class^="left"] { ... }`	`<span class="leftmost">Test</span>`
• Ends with:	`[class$="gap"] { ... }`	`<span class="rightgap">Test</span>`
• Contains:	`[class*="desc"] { ... }`	`<span class="description">Test</span>`

Combinator Selectors

- Descendant:

```
nav.content p { ... }
```

```
<nav class="content">
  <p>This element is selected</p>
  <div>
    <p>And so is this</p>
  </div>
</nav>
```

- Child selector:

```
main>h1  { ... }
```

```
<main>
  <h1>This element is selected</h1>
  <div>
    <h1>Not selected</h1>
  </div>
  <h1>Also selected</h1>
</nav>
```

Combinator Selectors

- Adjacent Sibling:

```
article#post+p { ... }
```

```
<article id="post">
  <p>Not selected</p>
</article>
<h2>Heading</h2>
<p>This is selected</p>
<p>Not selected</p>
```

- General Sibling:

```
article#post~p { ... }
```

```
<article id="post">
  <p>Not selected</p>
</article>
<h2>Heading</h2>
<p>This is selected</p>
<p>Selected as well</p>
```

Selectors

We explored the first 3 main element selection categories:

- Simple selectors (tag, id, class, universal, group, and their combination)
- Attribute selectors (by attribute name and value using exact and containing matching)
- Combinator selectors (descendant, direct children, adjacent and general siblings)
- Pseudo-class selectors
- Pseudo-element selectors

Pseudo-class and pseudo-elements will be explained later!

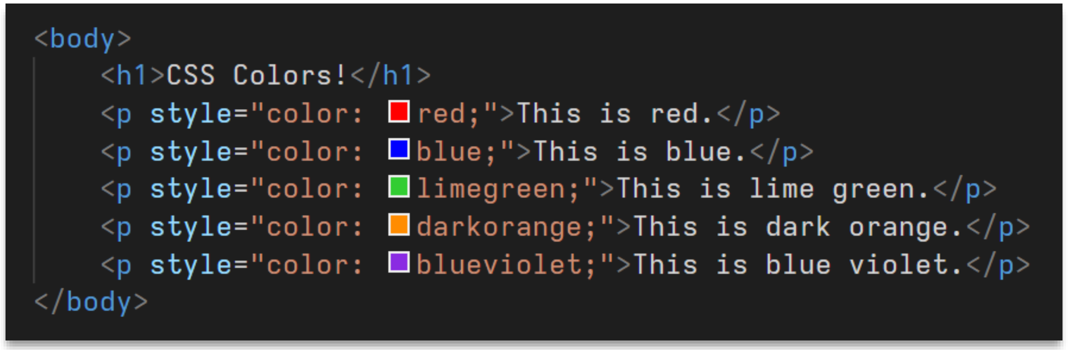
Color Properties

Colors

The first CSS property that we saw was the text **color** property.
The color property value can be a:

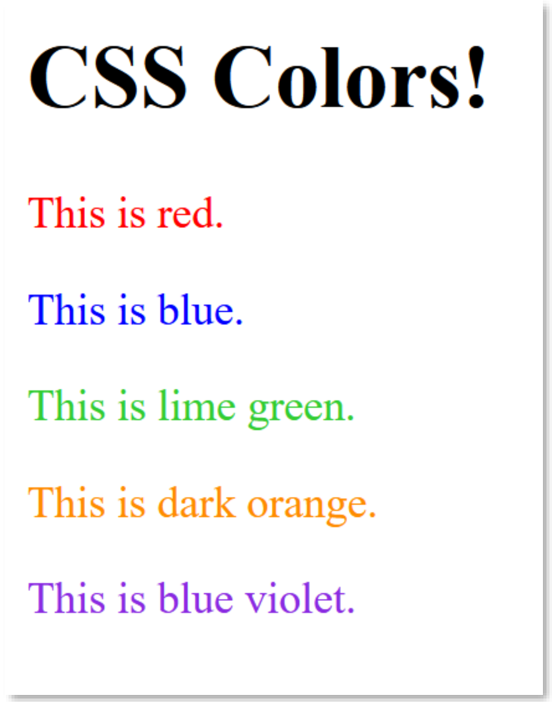
- Color name
- RGB (Red Green Blue) value
- Hex (Hexadecimal) value
- HSL (Hue Saturation Lightness) value

There are more than 150 standard color names supported by CSS.



```
<body>
    <h1>CSS Colors!</h1>
    <p style="color: 🟥red;">This is red.</p>
    <p style="color: 🟦blue;">This is blue.</p>
    <p style="color: 🟩limegreen;">This is lime green.</p>
    <p style="color: 🟧darkorange;">This is dark orange.</p>
    <p style="color: 🟪blueviolet;">This is blue violet.</p>
</body>
```

(Inline CSS is used here)



CSS Colors!

This is red.

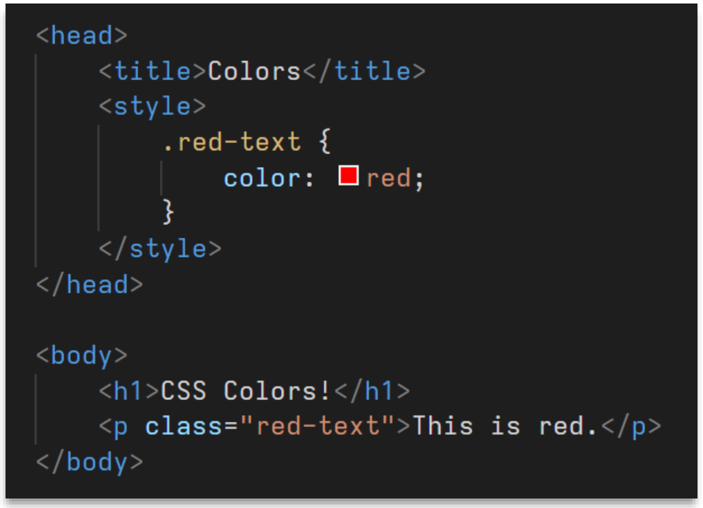
This is blue.

This is lime green.

This is dark orange.

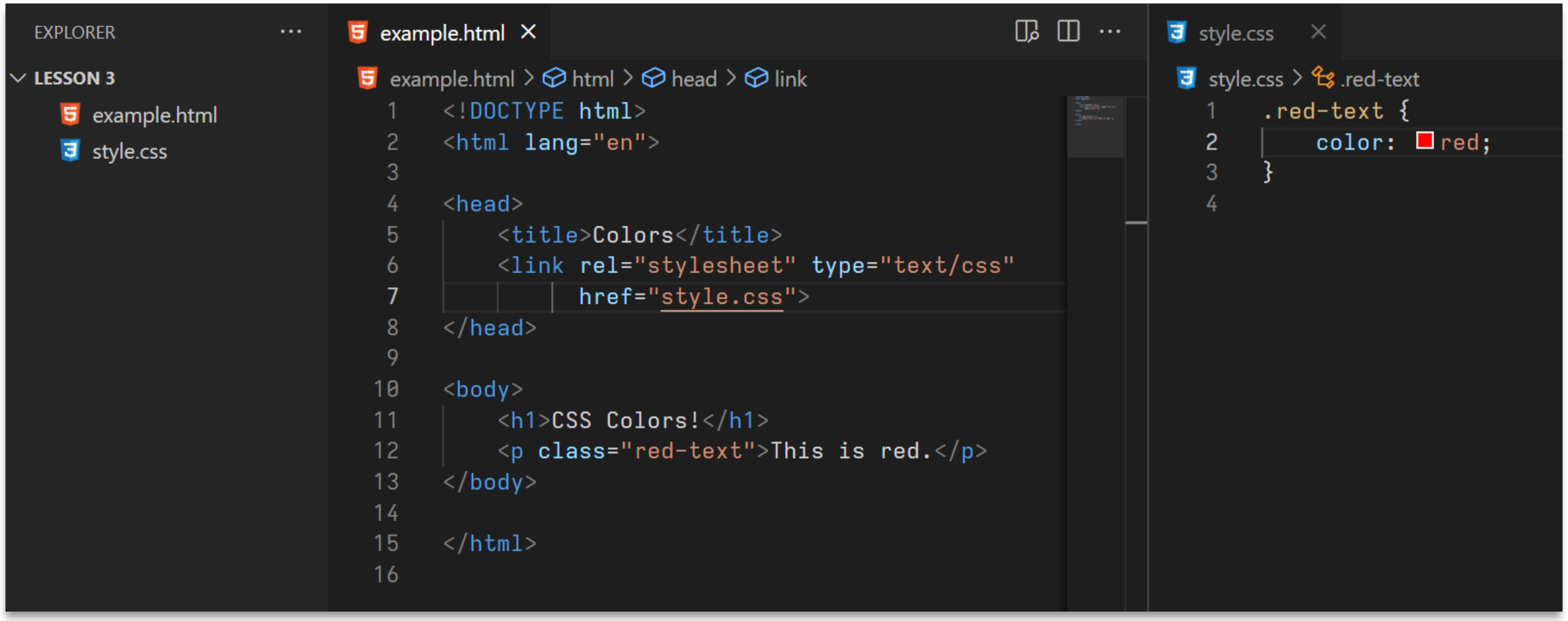
This is blue violet.

Colors



```
<head>
    <title>Colors</title>
    <style>
        .red-text {
            color: 🟥red;
        }
    </style>
</head>

<body>
    <h1>CSS Colors!</h1>
    <p class="red-text">This is red.</p>
</body>
```

(Internal CSS)



example.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Colors</title>
    <link rel="stylesheet" type="text/css"
        href="style.css">
</head>

<body>
    <h1>CSS Colors!</h1>
    <p class="red-text">This is red.</p>
</body>

</html>
```

style.css

```
.red-text {
    color: 🟥red;
}
```

(External CSS)

RGB

RGB color is made of 3 numbers in the range of 0 to 255: red, green, and blue.

`color: rgb(255 0 0);`	🟥	`color: rgb(255 255 0);`	🟨
`color: rgb(0 255 0);`	🟩	`color: rgb(0 255 255);`	🟦
`color: rgb(0 0 255);`	🟦	`color: rgb(255 0 255);`	🟪
`color: rgb(0 0 0);`	⬛	`color: rgb(200 200 200);`	■
`color: rgb(255 255 255);`	⬜	`color: rgb(40 40 40);`	⬛
`color: rgb(128 128 128);`	■	`color: rgb(52 154 250);`	🟦

RGB

The values can also be written using percentages.

```
color: rgb(100% 0 0);
```
🟥

```
color: rgb(50% 180 0);
```
🟩

The old way of writing used to have commas:

```
color: rgb(52, 154, 250);
```
🟦

The **rgba** keyword is the same as **rgb**.

```
color: rgba(52 154 250);
```
🟦

Alpha

There can be a forth value indicating the color's **opacity**, also known as the **alpha** value.
The value is between 0 and 1 (or 0% and 100%).
The default value is 1 (the color is fully **opaque**).
A value of 0 makes the color completely **transparent** and invisible.

`color: rgb(255 0 0 / 1);`	🟥	`color: rgb(255 0 0 / 0);`	⬜
`color: rgb(255 0 0 / .5);`	🟥	`color: transparent;`	⬜
`color: rgb(255 0 0 / 20%);`	🟥		

In the old way of writing, a forth comma was used:

`color: rgb(52, 154, 250, 1);`	🟦

Hex

Another way of writing RGB colors is to use a hex value.
Hex values are a way of encoding numbers using the following table:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Decimal system: 42 = 4x10 + 2 = 42
Hexadecimal system: 42 = 4x16 + 2 = 66,
4A = 4x16 + A = 64 + 10 = 74

The highest 2 digit hex number is FF which is equal to: Fx16 + F = 15x16 + 15 = 255
So each RGB value can be represented using a 2 digit hex value.

Hex

The RGB values are written as: #RRGGBB

`color: rgb(255 0 0);`	🟥	`color: #ff0000;`	🟥
`color: rgb(0 255 0);`	🟩	`color: #00ff00;`	🟩
`color: rgb(52 154 250);`	🟦	`color: #349afa;`	🟦
`color: rgb(0 0 0);`	⬛	`color: #000000;`	⬛
`color: rgb(255 255 255);`	⬜	`color: #ffffff;`	⬜

Hex

RGBA values are written as: #RRGGBBAA

```
color: rgb(255 0 0 / .5);
```

🟥

```
color: #ff000080
```

🟥

We can use a shorthand if all RGB hex pairs are the same:

```
color: #ff0000
```

🟥

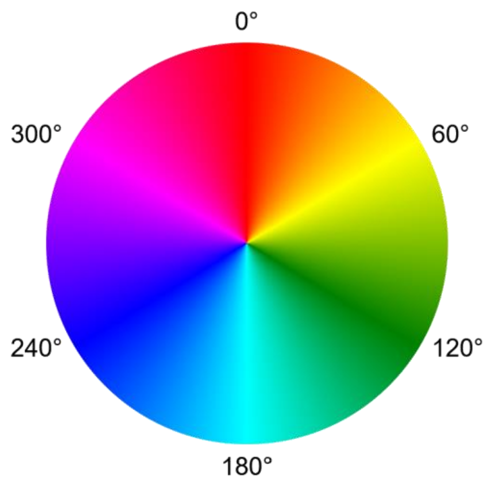
```
color: #f00
```

🟥

HSL

The HSL color model represents the same color space as RGB does (sRGB color space).
Hue: this is the actual color element between 0 and 360 (can optionally have the **deg** unit e.g. 360deg).
Saturation: a value between 0% and 100% representing the color's saturation.
Lightness: a value between 0% and 100% representing the color's brightness.
Just like RGB, an additional alpha parameter can also be present.

`color: rgb(255 0 0);`	🟥	`color: hsl(0 100% 50%);`	🟥
`color: rgb(0 255 0);`	🟩	`color: hsl(120 100% 50%);`	🟩
`color: rgb(0 0 255);`	🟦	`color: hsl(240 100% 50%);`	🟦
`color: rgb(0 0 0);`	⬛	`color: hsl(0 0% 0%);`	⬛
`color: rgb(255 255 255);`	⬜	`color: hsl(0 0% 100%);`	⬜



HSL

With alpha:

```
color: rgb(255 0 0 / .5);
```
🟥

```
color: hsl(0 100% 50% / .5);
```
🟥

```
color: rgb(52 154 250);
```
🟦

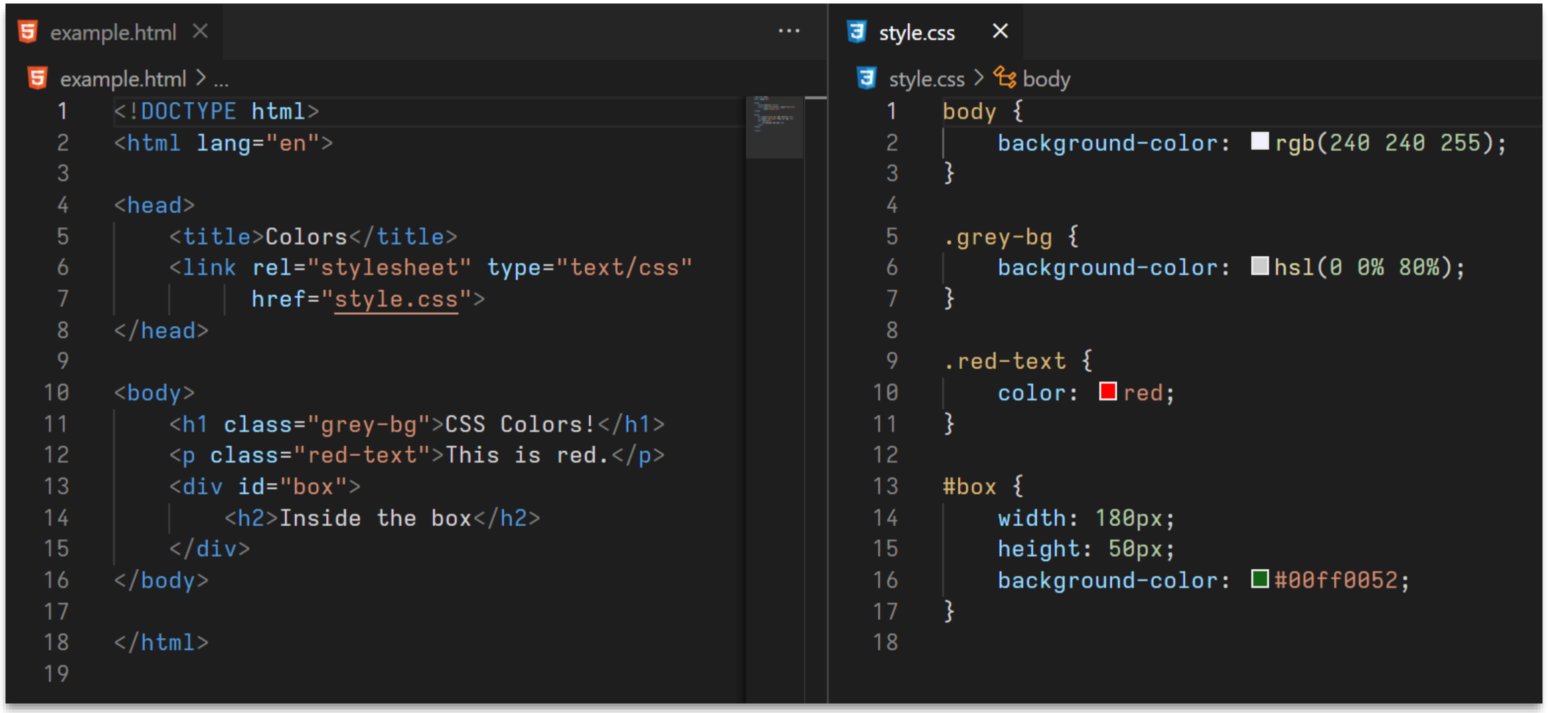
```
color: hsl(209 95% 59%);
```
🟦

HSL

Easily change the brightness or the saturation without changing the base color.

		`color: hsl(120 100% 80%);`	🟩
`color: rgb(0 255 0);`	🟩 ←	`color: hsl(120 100% 50%);`	🟩
		`color: hsl(120 100% 20%);`	🟩
		`color: hsl(120 50% 50%);`	🟩
`color: rgb(102 153 102);`	🟩 ←	`color: hsl(120 20% 50%);`	🟩
		`color: hsl(120 0% 50%);`	⬛

Background Color

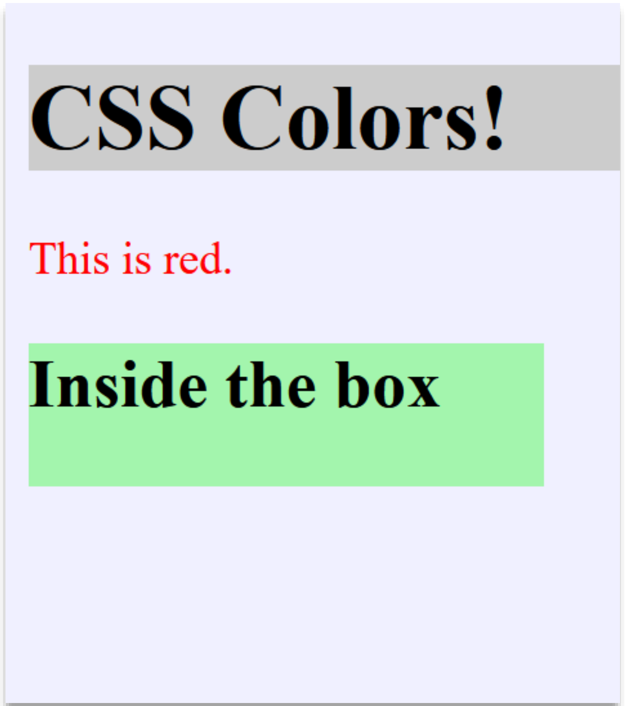


example.html × | example.html > ...

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Colors</title>
    <link rel="stylesheet" type="text/css"
        href="style.css">
</head>

<body>
    <h1 class="grey-bg">CSS Colors!</h1>
    <p class="red-text">This is red.</p>
    <div id="box">
        <h2>Inside the box</h2>
    </div>
</body>

</html>

```

style.css × | style.css > body

```css
body {
    background-color: rgb(240 240 255);
}

.grey-bg {
    background-color: hsl(0 0% 80%);
}

.red-text {
    color: red;
}

#box {
    width: 180px;
    height: 50px;
    background-color: #00ff0052;
}

```



CSS Colors!

This is red.

Inside the box

Text Formatting Properties

Text Formatting

We have already seen the **color** property in detail.
The **color** property sets the text color, and the **background-color** property sets its background.

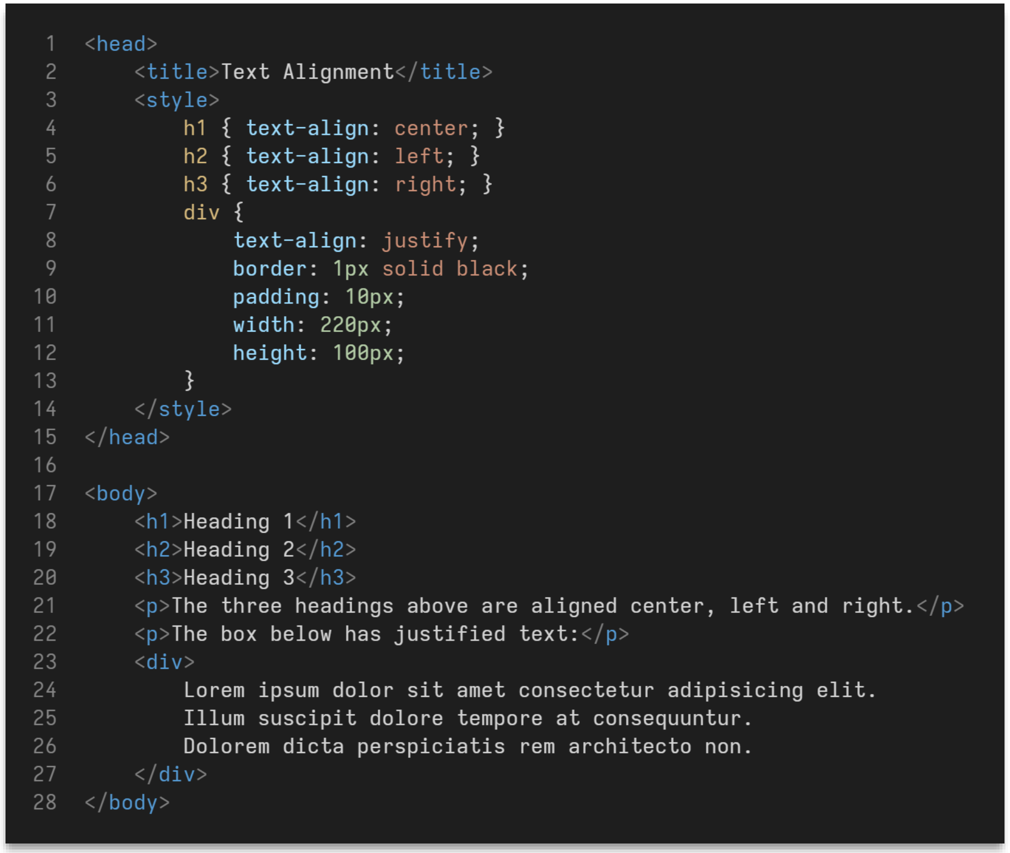
Here we are going to take a look at the following properties:

- text-align
- direction
- text-decoration
- text-transform
- text-indent
- letter-spacing
- word-spacing
- line-height
- text-shadow

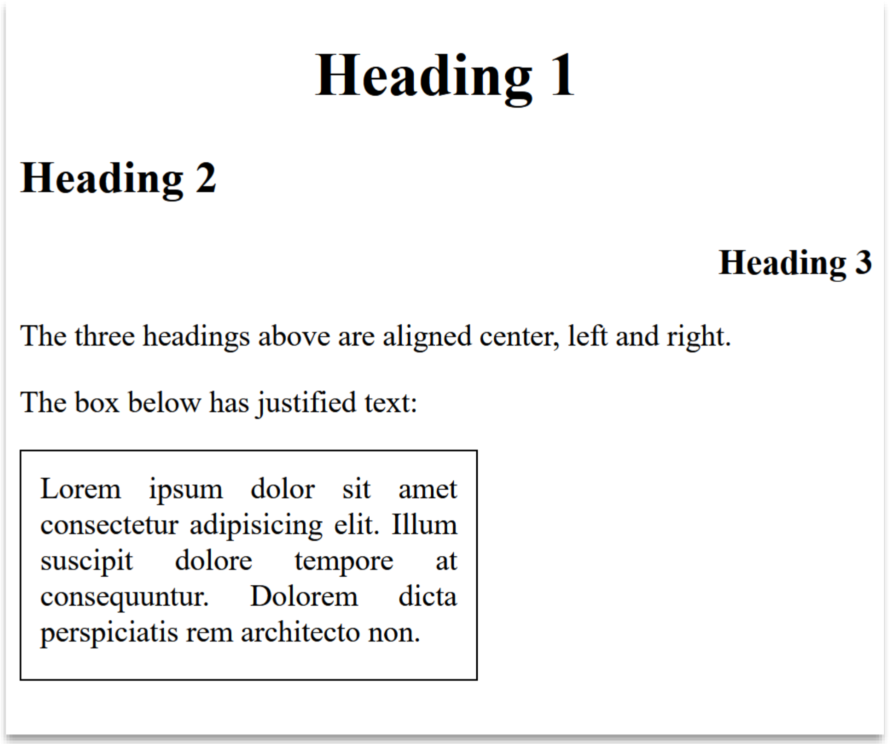
- font-size
- font-style
- font-weight

Text Alignment

The **text-align** property sets the horizontal alignment of text. (**text-align-last** aligns the last line of a text)
Valid values: left, right, center, justify.



```html
<head>
    <title>Text Alignment</title>
    <style>
        h1 { text-align: center; }
        h2 { text-align: left; }
        h3 { text-align: right; }
        div {
            text-align: justify;
            border: 1px solid black;
            padding: 10px;
            width: 220px;
            height: 100px;
        }
    </style>
</head>

<body>
    <h1>Heading 1</h1>
    <h2>Heading 2</h2>
    <h3>Heading 3</h3>
    <p>The three headings above are aligned center, left and right.</p>
    <p>The box below has justified text:</p>
    <div>
        Lorem ipsum dolor sit amet consectetur adipisicing elit.
        Illum suscipit dolore tempore at consequuntur.
        Dolorem dicta perspiciatis rem architecto non.
    </div>
</body>
```



Heading 1

Heading 2

Heading 3

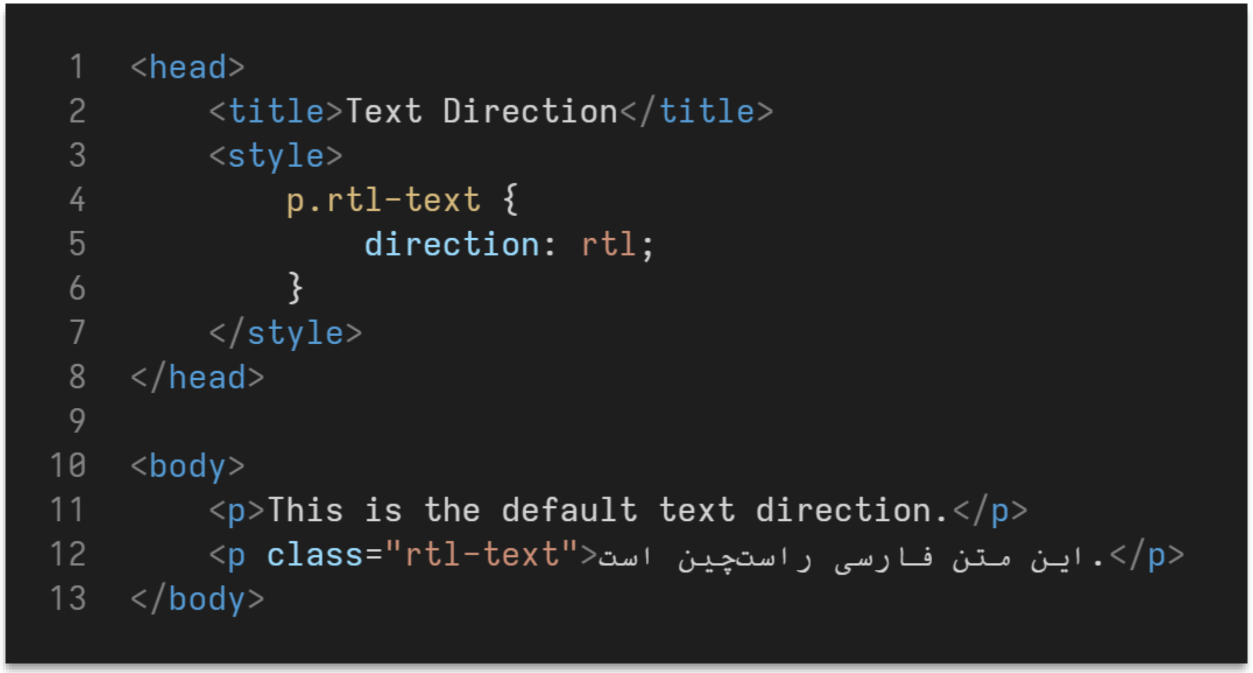
The three headings above are aligned center, left and right.

The box below has justified text:

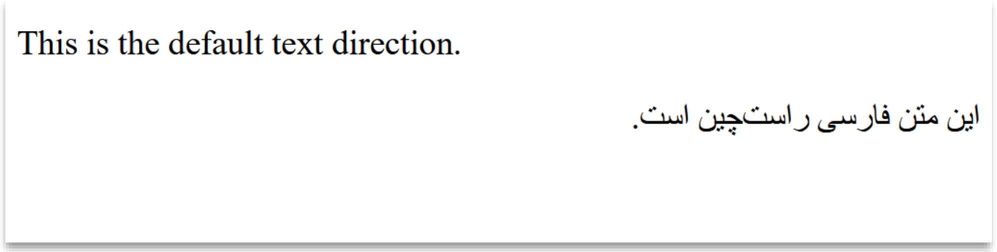
Lorem ipsum dolor sit amet consectetur adipisicing elit. Illum suscipit dolore tempore at consequuntur. Dolorem dicta perspiciatis rem architecto non.

Text Direction

The **direction** property sets the text direction. Valid values: rtl, ltr.
This property should usually be avoided and the HTML **dir** attribute should be used instead.



```
<head>
    <title>Text Direction</title>
    <style>
        p.rtl-text {
            direction: rtl;
        }
    </style>
</head>

<body>
    <p>This is the default text direction.</p>
    <p class="rtl-text">این متن فارسی راستچین است.</p>
</body>
```



This is the default text direction.

این متن فارسی راستچین است.

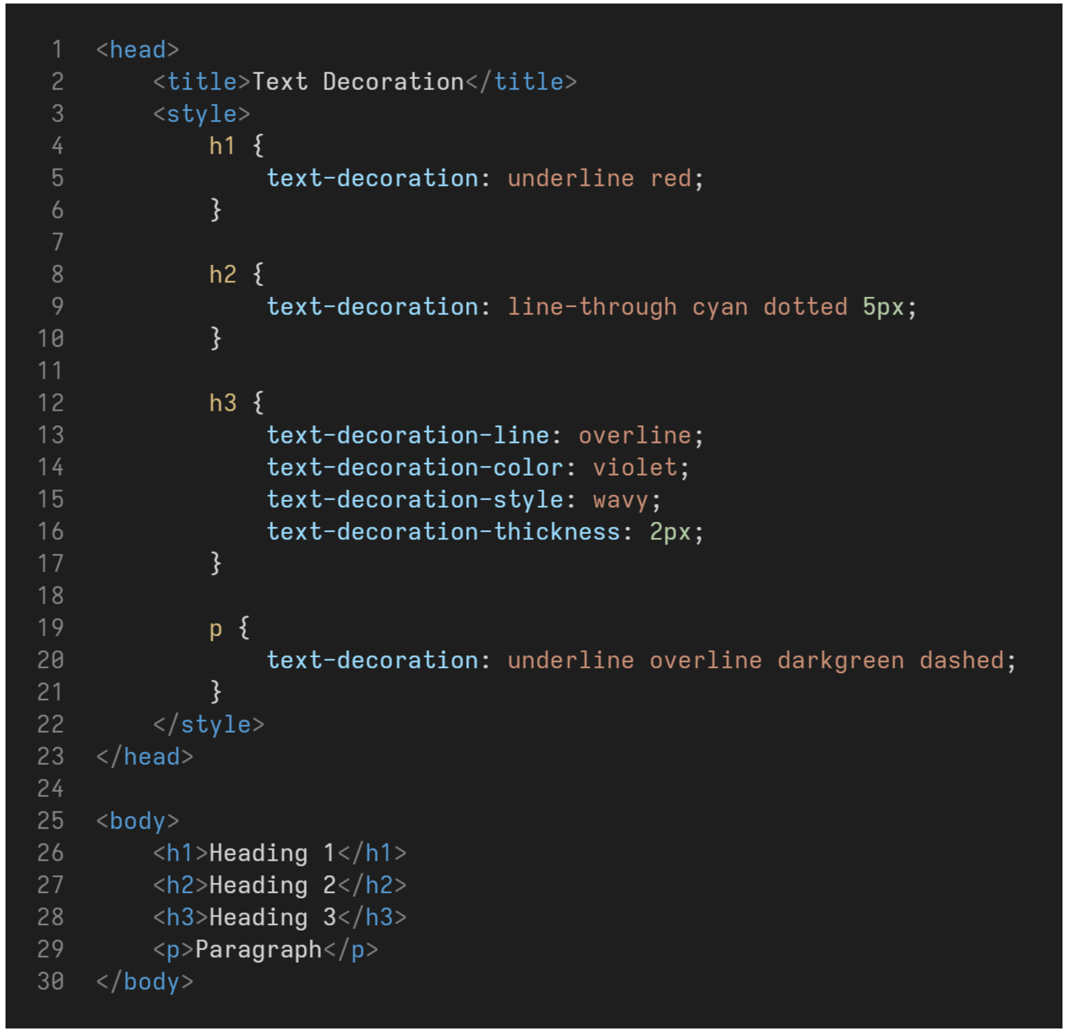
Text Decoration

The **text-decoration** property allows setting different decorations such as underlines to text.
The property is a **shorthand property**, meaning that there are sub-properties that can be set separately, and the shorthand property allows setting them all in one property.

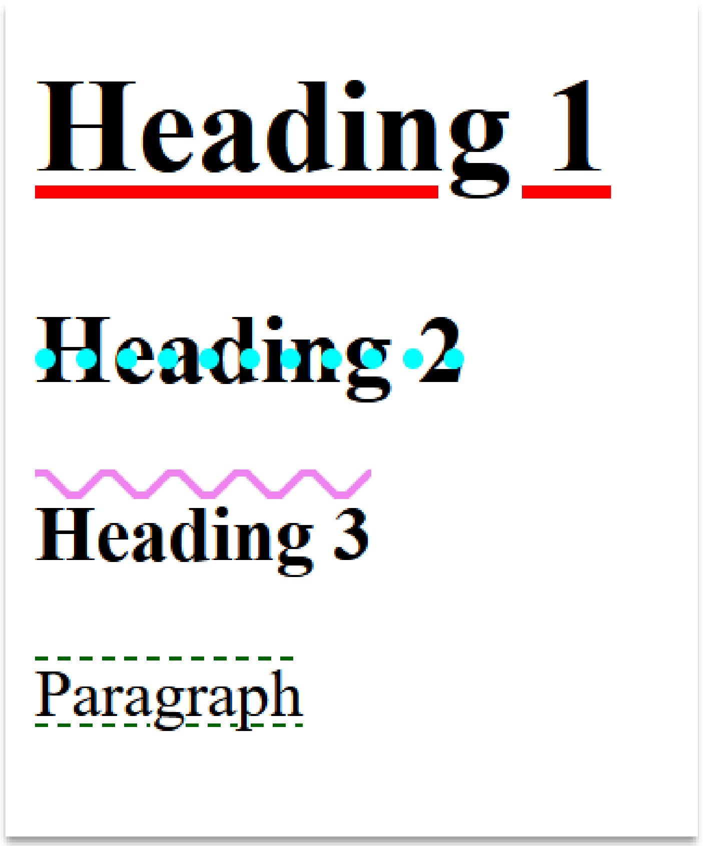
```
text-decoration = text-decoration-line text-decoration-color
                  text-decoration-style text-decoration-thickness
```

- Valid decoration lines: none, overline, line-through, underline, or a list of them.
- Valid decoration colors are any valid CSS colors.
- Valid decoration styles: solid, double, dotted, dashed, wavy.
- Valid decoration thickness is any valid CSS length unit (such as px).

Text Decoration



```html
<head>
    <title>Text Decoration</title>
    <style>
        h1 {
            text-decoration: underline red;
        }

        h2 {
            text-decoration: line-through cyan dotted 5px;
        }

        h3 {
            text-decoration-line: overline;
            text-decoration-color: violet;
            text-decoration-style: wavy;
            text-decoration-thickness: 2px;
        }

        p {
            text-decoration: underline overline darkgreen dashed;
        }
    </style>
</head>

<body>
    <h1>Heading 1</h1>
    <h2>Heading 2</h2>
    <h3>Heading 3</h3>
    <p>Paragraph</p>
</body>
```



Heading 1

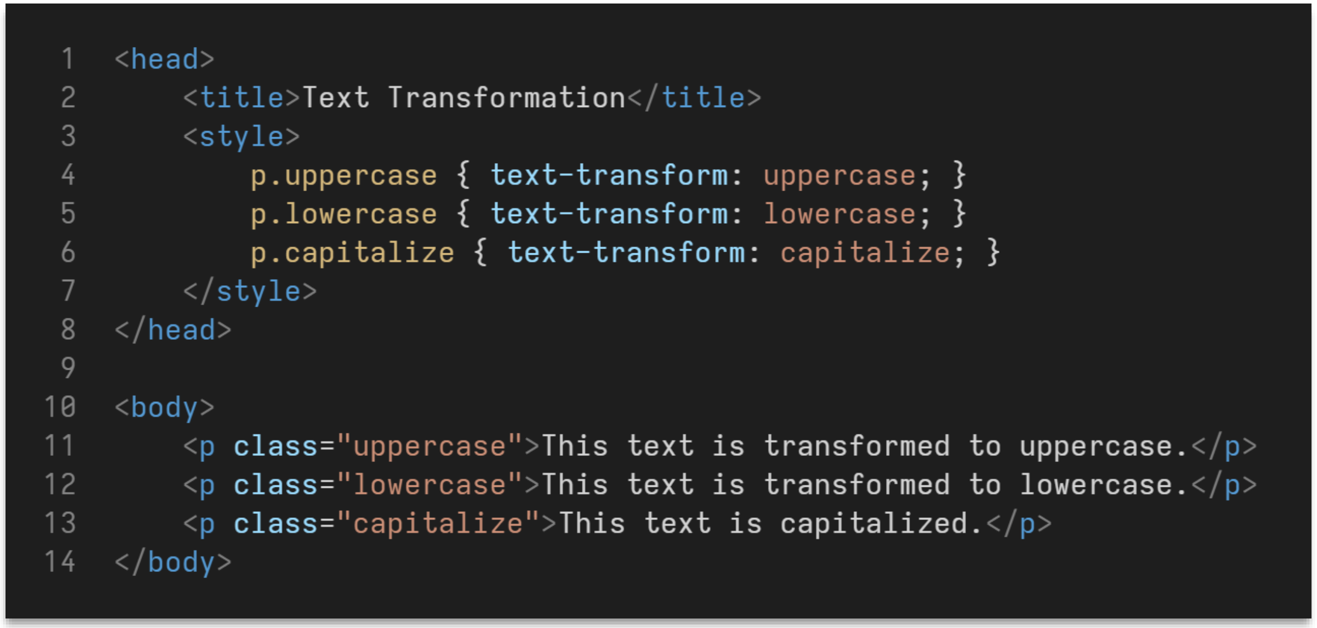
Heading 2

Heading 3

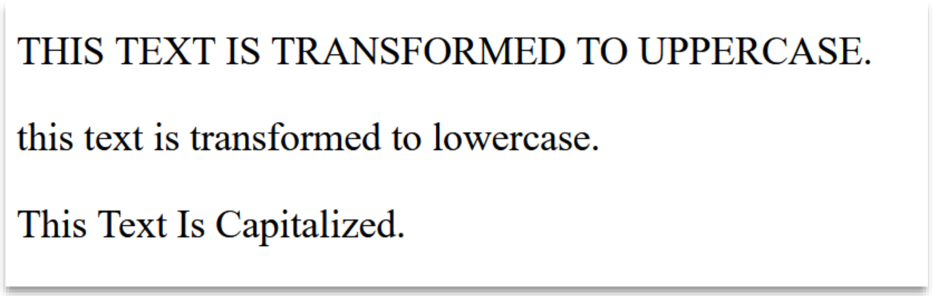
Paragraph

Text Transformation

The **text-transform** property controls the capitalization of text.
Valid values: uppercase, lowercase, capitalize.



```html
<head>
    <title>Text Transformation</title>
    <style>
        p.uppercase { text-transform: uppercase; }
        p.lowercase { text-transform: lowercase; }
        p.capitalize { text-transform: capitalize; }
    </style>
</head>

<body>
    <p class="uppercase">This text is transformed to uppercase.</p>
    <p class="lowercase">This text is transformed to lowercase.</p>
    <p class="capitalize">This text is capitalized.</p>
</body>
```



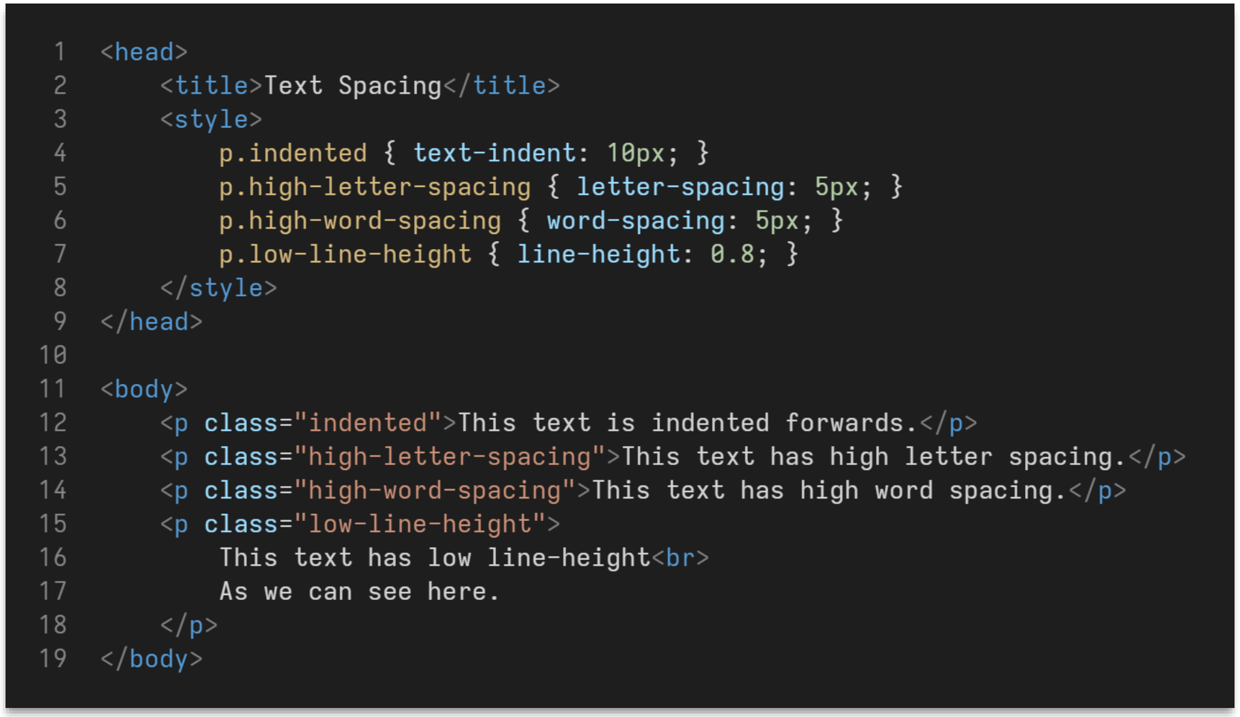
THIS TEXT IS TRANSFORMED TO UPPERCASE.

this text is transformed to lowercase.

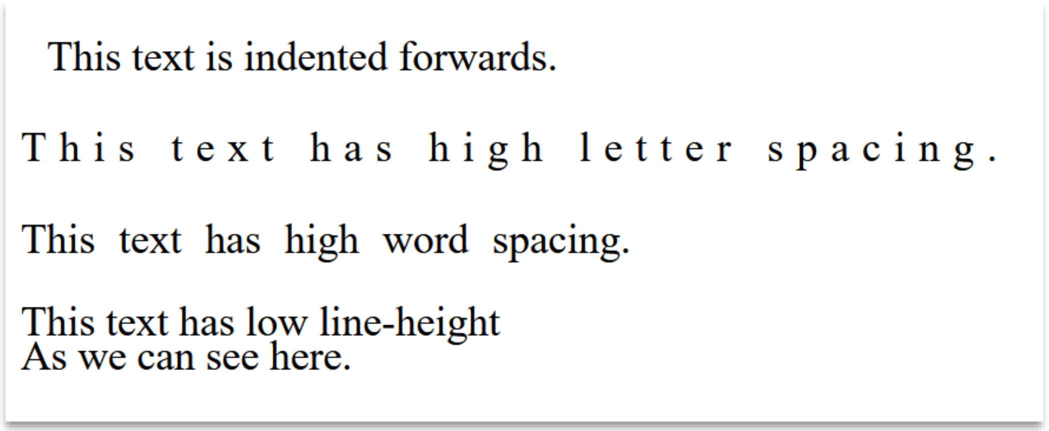
This Text Is Capitalized.

Text Spacing

The **text-indent** property specifies the indentation of the first line of a text.
The **letter-spacing** property specifies the space between the characters of a text.
The **word-spacing** property specifies the space between the words of a text.
The **line-height** property specifies the space between lines.



```html
<head>
    <title>Text Spacing</title>
    <style>
        p.indented { text-indent: 10px; }
        p.high-letter-spacing { letter-spacing: 5px; }
        p.high-word-spacing { word-spacing: 5px; }
        p.low-line-height { line-height: 0.8; }
    </style>
</head>

<body>
    <p class="indented">This text is indented forwards.</p>
    <p class="high-letter-spacing">This text has high letter spacing.</p>
    <p class="high-word-spacing">This text has high word spacing.</p>
    <p class="low-line-height">
        This text has low line-height<br>
        As we can see here.
    </p>
</body>
```



This text is indented forwards.

T h i s   t e x t   h a s   h i g h   l e t t e r   s p a c i n g .

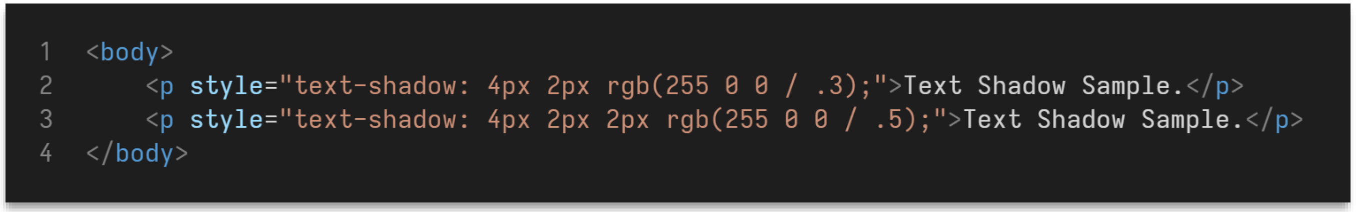
This text has high word spacing.

This text has low line-height
As we can see here.

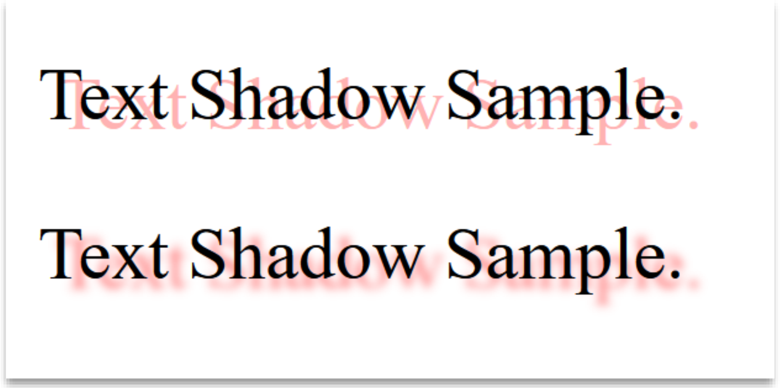
Text Shadow

The **text-shadow** property adds shadow to text.

`text-shadow: horizontal-offset vertical-offset [optional]blur color`



```html
<body>
    <p style="text-shadow: 4px 2px rgb(255 0 0 / .3);">Text Shadow Sample.</p>
    <p style="text-shadow: 4px 2px 2px rgb(255 0 0 / .5);">Text Shadow Sample.</p>
</body>
```

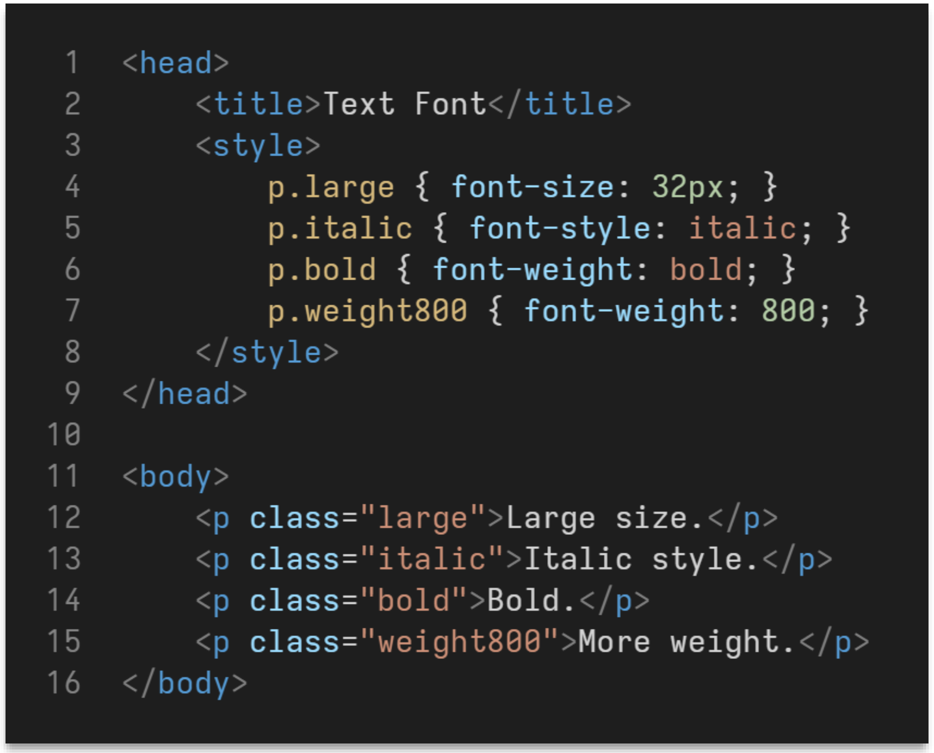


Text Shadow Sample.

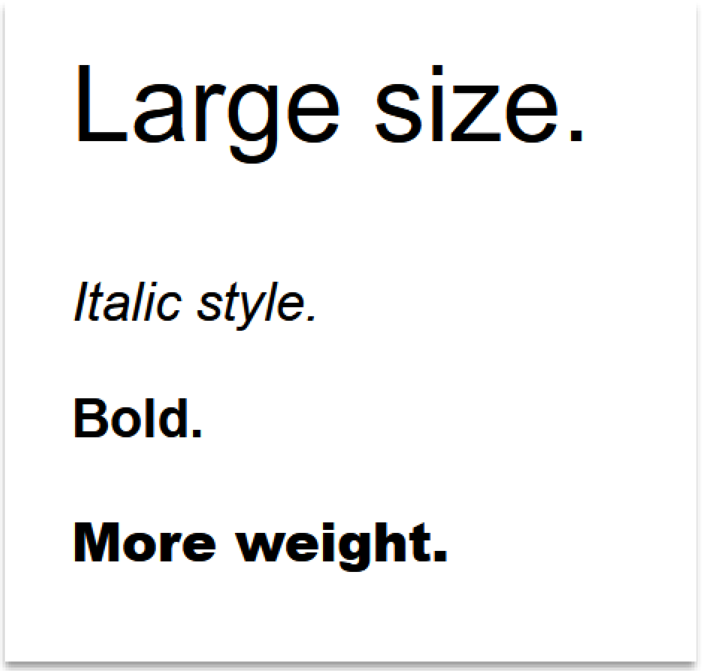
Text Shadow Sample.

Font

The **font-size** property sets the font size (units such as px, em, rem, %, vw).
The **font-style** property can set the text to be italic.
The **font-weight** property sets the weight of the font (100-900). 400 is normal, 700 is bold.



```html
<head>
    <title>Text Font</title>
    <style>
        p.large { font-size: 32px; }
        p.italic { font-style: italic; }
        p.bold { font-weight: bold; }
        p.weight800 { font-weight: 800; }
    </style>
</head>

<body>
    <p class="large">Large size.</p>
    <p class="italic">Italic style.</p>
    <p class="bold">Bold.</p>
    <p class="weight800">More weight.</p>
</body>
```



Large size.

Italic style.

Bold.

More weight.

Box Model

Outer Display Type

All HTML elements can be considered as boxes.

If a box has an outer display type of **block** (such as h1, p, div, …):

- It will take its own new line.
- width and height properties are respected.
- Margin, padding and borders cause other elements to be pushed away from the box.

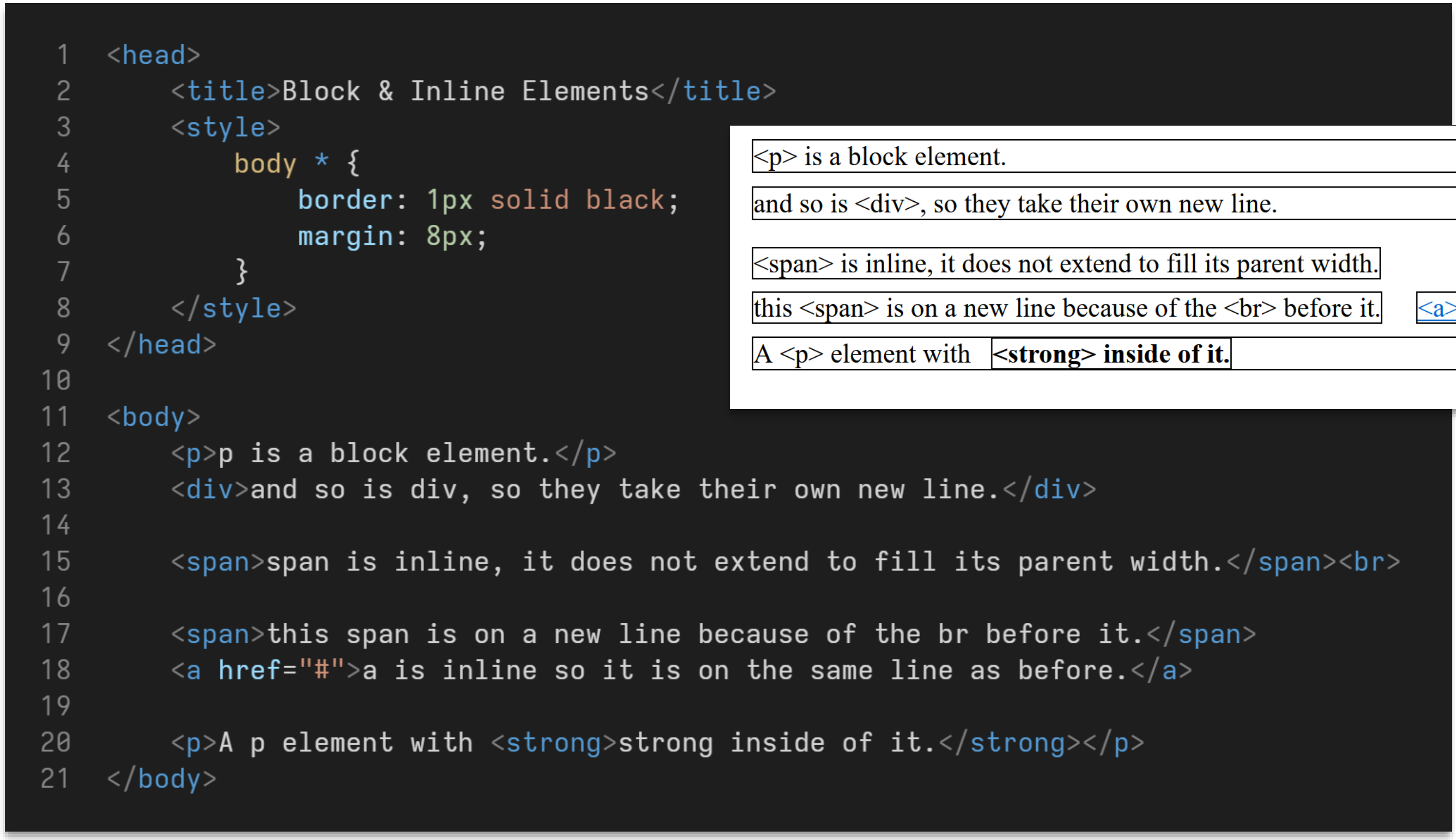
These are not the case with **inline** elements (such as span, a, strong, …).

The CSS display property can be used to set the outer and inner display type:

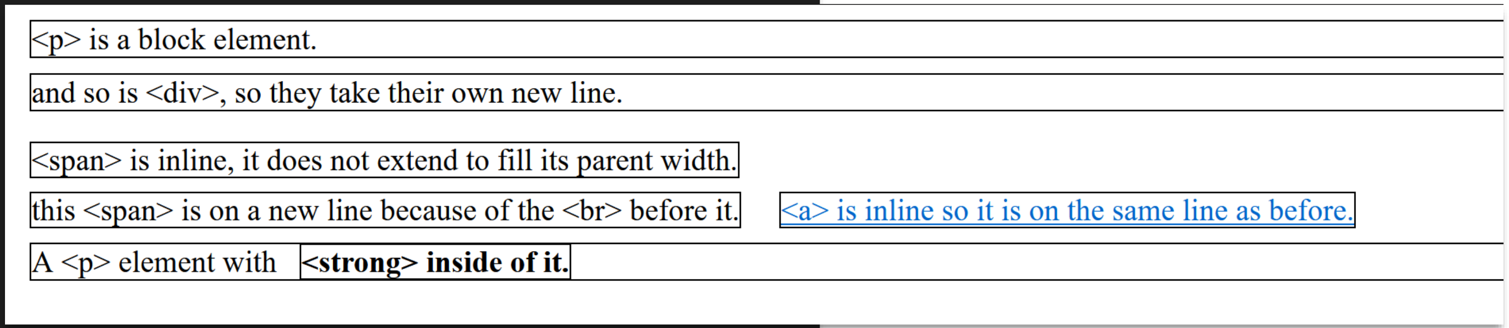
```css
display: block;
display: inline;
```

So for example, **div** has display: block and **span** has display: inline as the default outer display type.
The inner display types are also these 2 based on the tag but can be set to other types such as **flex** and **grid**.

Outer Display Type



```html
<head>
    <title>Block & Inline Elements</title>
    <style>
        body * {
            border: 1px solid black;
            margin: 8px;
        }
    </style>
</head>

<body>
    <p>p is a block element.</p>
    <div>and so is div, so they take their own new line.</div>

    <span>span is inline, it does not extend to fill its parent width.</span><br>

    <span>this span is on a new line because of the br before it.</span>
    <a href="#">a is inline so it is on the same line as before.</a>

    <p>A p element with <strong>strong inside of it.</strong></p>
</body>
```



<p> is a block element.

and so is <div>, so they take their own new line.

<span> is inline, it does not extend to fill its parent width.

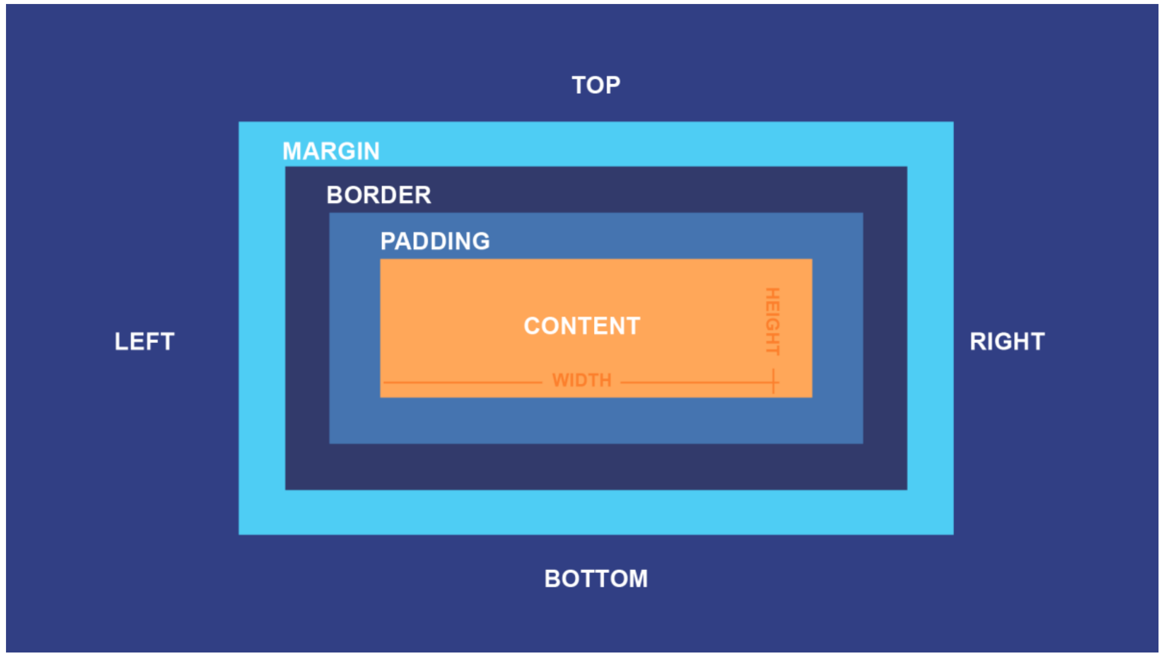
this <span> is on a new line because of the <br> before it. <a> is inline so it is on the same line as before.

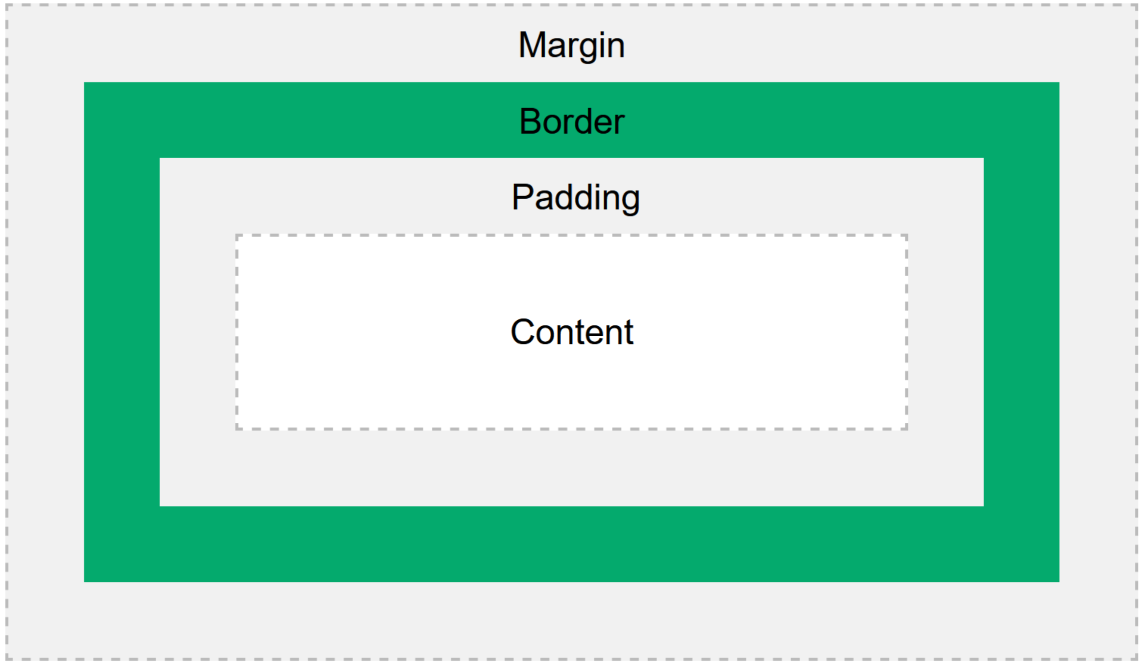
A <p> element with **<strong> inside of it.**

Box Model

The box model is used by block elements to define how **margin**, **border**, **padding** and **content** work together.

- Content box: this is the content such as the text inside a <p> tag.
- Padding box: padding sits around the content as white space.
- Border box: the border wraps the padding and content.
- Margin box: the margin wraps everything and is placed between other elements and our border.

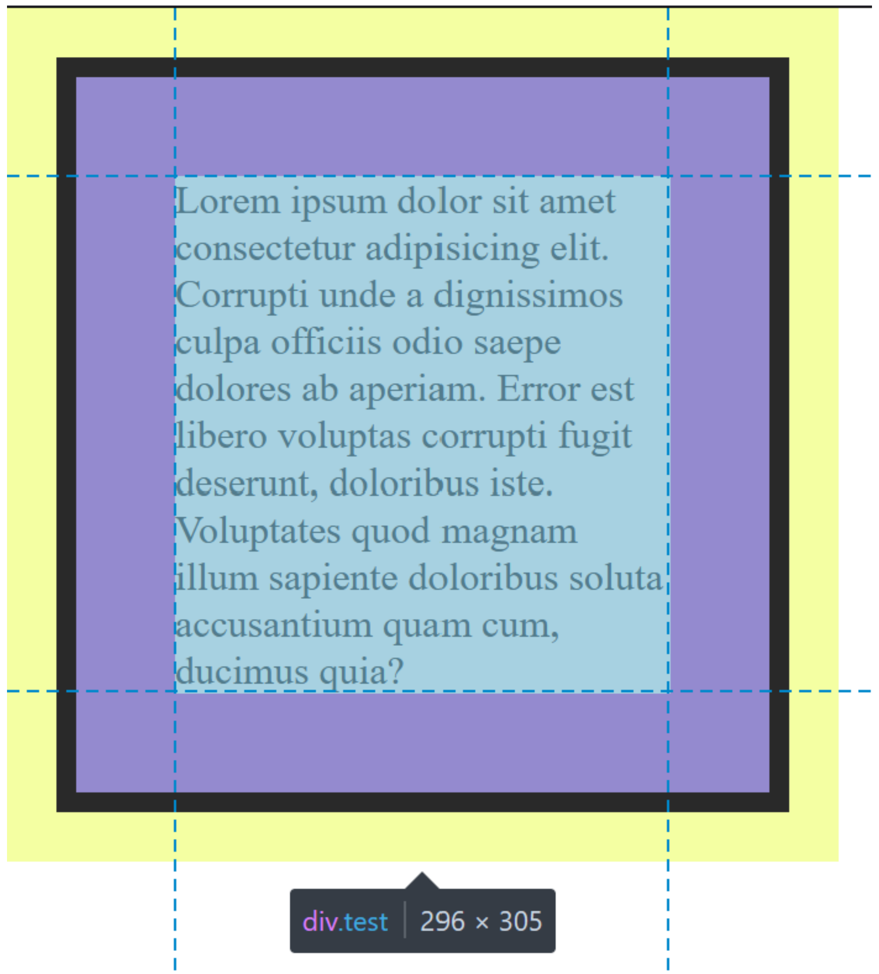




Box Model

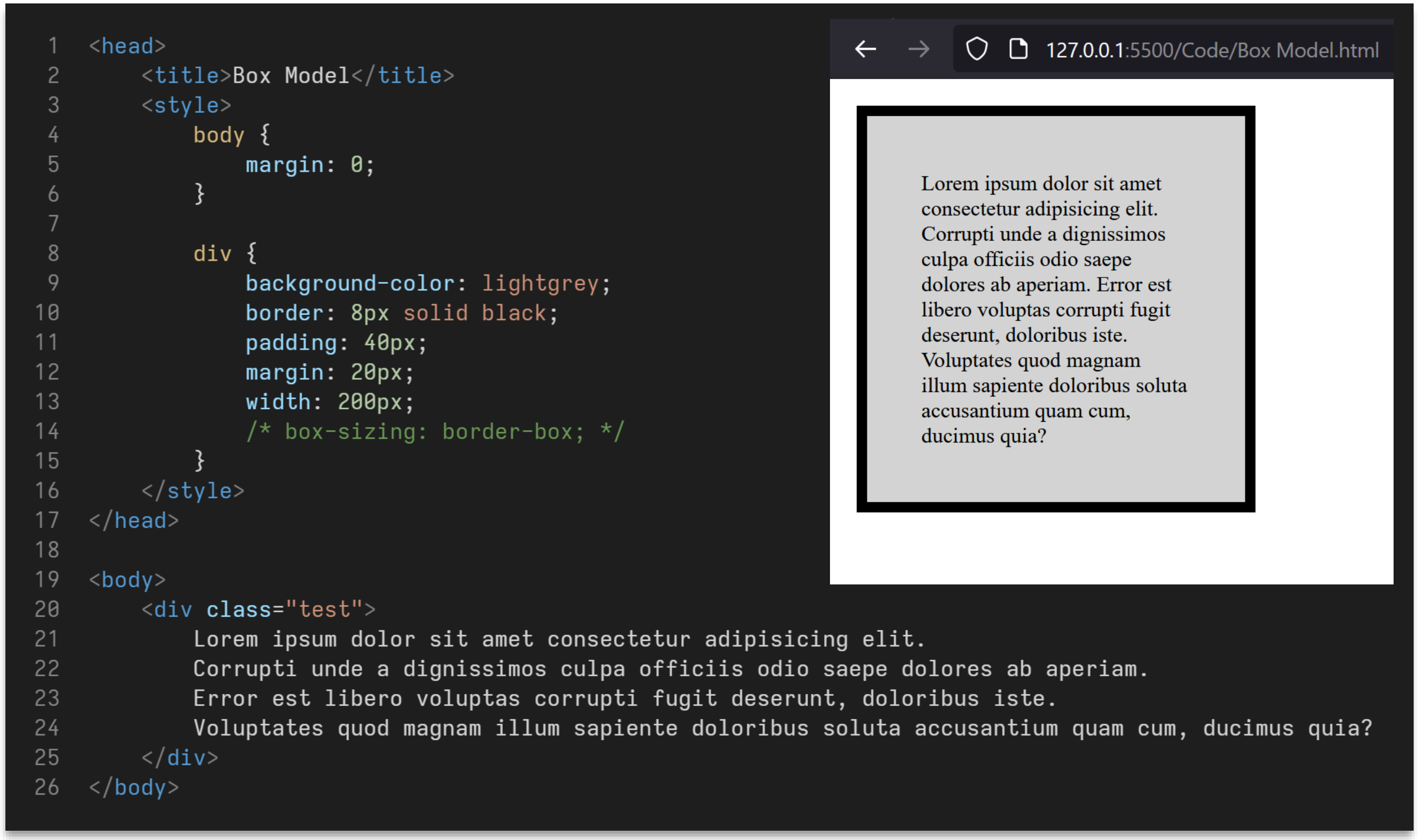
We can see this more clearly in the browser developer tools. (Right-click and Inspect)

- Content box: blue and grid lines.
- Padding box: purple.
- Border box: black.
- Margin box: yellow.



Lorem ipsum dolor sit amet consectetur adipisicing elit. Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam. Error est libero voluptas corrupti fugit deserunt, doloribus iste. Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?

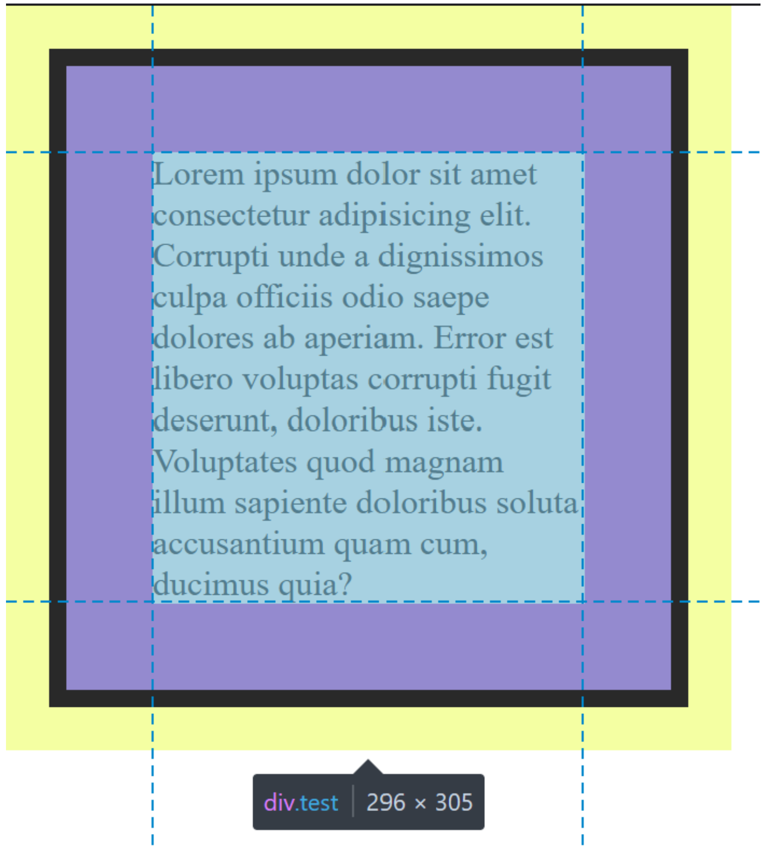
Box Model



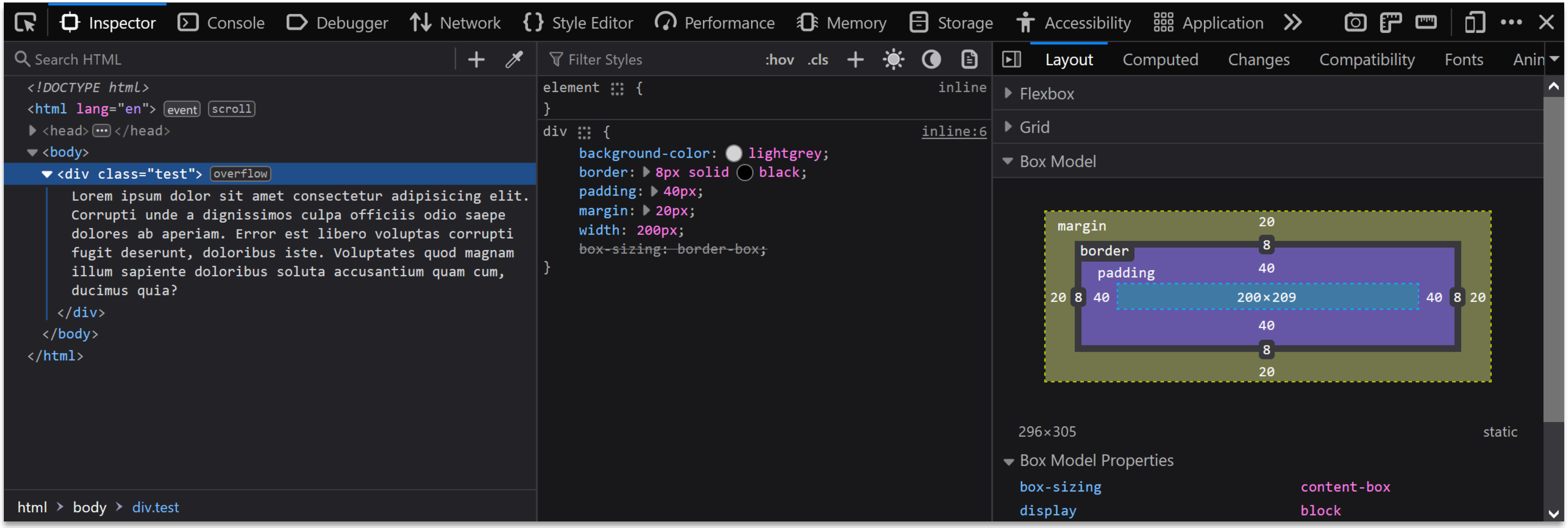
```html
<head>
    <title>Box Model</title>
    <style>
        body {
            margin: 0;
        }

        div {
            background-color: lightgrey;
            border: 8px solid black;
            padding: 40px;
            margin: 20px;
            width: 200px;
            /* box-sizing: border-box; */
        }
    </style>
</head>

<body>
    <div class="test">
        Lorem ipsum dolor sit amet consectetur adipisicing elit.
        Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam.
        Error est libero voluptas corrupti fugit deserunt, doloribus iste.
        Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?
    </div>
</body>
```

127.0.0.1:5500/Code/Box Model.html

Lorem ipsum dolor sit amet consectetur adipisicing elit. Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam. Error est libero voluptas corrupti fugit deserunt, doloribus iste. Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?


Lorem ipsum dolor sit amet consectetur adipisicing elit. Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam. Error est libero voluptas corrupti fugit deserunt, doloribus iste. Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?
div.test
296 × 305

Box Model



```
<!DOCTYPE html>
<html lang="en"> event scroll
  <head>…</head>
  <body>
    <div class="test"> overflow
      Lorem ipsum dolor sit amet consectetur adipisicing elit.
      Corrupti unde a dignissimos culpa officiis odio saepe
      dolores ab aperiam. Error est libero voluptas corrupti
      fugit deserunt, doloribus iste. Voluptates quod magnam
      illum sapiente doloribus soluta accusantium quam cum,
      ducimus quia?
    </div>
  </body>
</html>
```

html > body > div.test

```
element {                                   inline
}
div {                                     inline:6
    background-color: lightgrey;
    border: 8px solid black;
    padding: 40px;
    margin: 20px;
    width: 200px;
    box-sizing: border-box;
}
```

Box model: margin 20 / 20 / 20 / 20; border 8 / 8 / 8 / 8; padding 40 / 40 / 40 / 40; content 200×209

296×305 static

Box Model Properties

box-sizing	content-box
display	block

Box Sizing

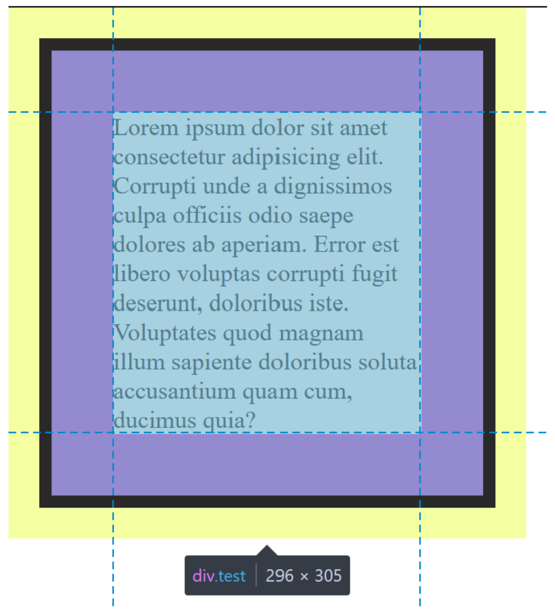
The width and height properties set the content box size by default.
This means that padding and border add to the overall size of the box.
Usually we want to set the width and height of the whole box.
This means that the default box-sizing of content-box is not wanted.

Both boxes have a width of 200px.

Left box: the content-box is 200px which has padding and border added to it making the box 296px.
200px (content) + 40px + 40px (left/right padding) + 8px + 8px (left/right border)

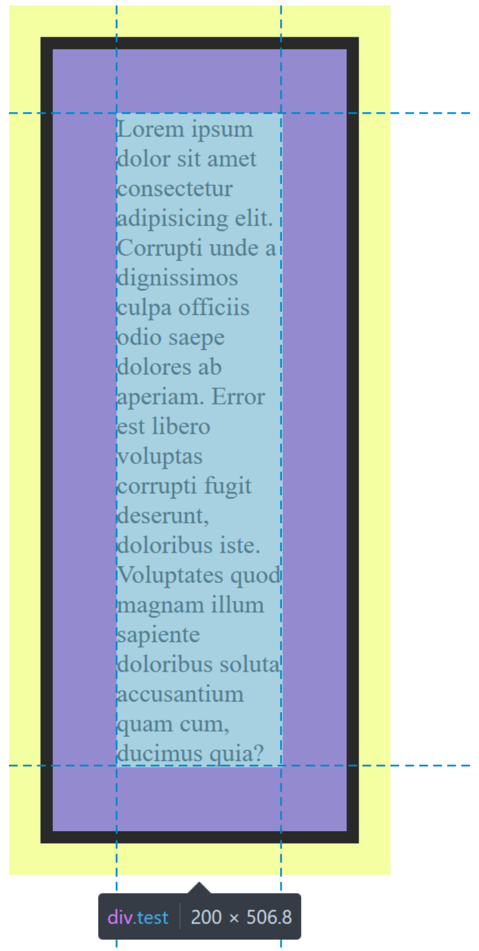
Right box: the border-box is 200px which means the content is lowered (104px) so when the padding and border thickness is added, the whole box is 200px.
104px (content) + 80px (padding) + 16px (border)

```
box-sizing: content-box;
```



Lorem ipsum dolor sit amet consectetur adipisicing elit. Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam. Error est libero voluptas corrupti fugit deserunt, doloribus iste. Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?

```
box-sizing: border-box;
```



Lorem ipsum dolor sit amet consectetur adipisicing elit. Corrupti unde a dignissimos culpa officiis odio saepe dolores ab aperiam. Error est libero voluptas corrupti fugit deserunt, doloribus iste. Voluptates quod magnam illum sapiente doloribus soluta accusantium quam cum, ducimus quia?

Padding

Padding is a shorthand property:

`padding: padding-top padding-right padding-bottom padding-left`

The 4 properties set the padding for each side.
The complete shorthand form takes all 4 properties in a clockwise direction from the top.

`padding: 1px 2px 3px 4px;`

There are also shorter forms:

```css
padding: 2px; /* sets 2px for all 4 sides */
padding: 2px 4px; /* sets 2px top & bottom, 4px for left & right */
padding: 2px 3px 4px; /* sets 2px for top, 3px for left & right, 4px for bottom */
padding: 1px 2px 3px 4px; /* sets each side separately */
```

Margin

The margin property is almost the same as padding and the takes such values:

```
margin: 2px; /* sets 2px for all 4 sides */
margin: 2px 4px; /* sets 2px top & bottom, 4px for left & right */
margin-top: 3px; /* sets 3px for top */
```

Border

Border is a shorthand property.

```
border: border-width border-style border-color
```

- border-width is any acceptable length unit.
- border-style can be solid, dotted, dashed, double, ...
- border-color is any color value (name, rgb, hex, hsl)

```
border: 2px solid blue;
```

To set border for specific sides or different borders for each side, the longhand properties should be used. For example, border-width-left.

Thank you for your attention.