LESSON #1 CASCADING STYLE SHEET (CSS)

BY: DR. PAULINE ABESAMIS

OCTOBER 8, 2024

INTRODUCTION TO CSS

- CSS is the acronym of "Cascading Style Sheets". CSS is **a computer language for laying out and structuring web pages (HTML or XML)**.
- CSS allows you to create rules that specify how the content of an element should appear.
- For example, you can specify that the background of the page is cream, all paragraphs should appear in gray using the Arial typeface, or that all level one headings should be in a blue, italic, Times typeface.
- Browser support has been quite inconsistent, and significant bugs, particularly in older of versions of Internet Explorer, have made the use of CSS a lesson in frustration.

CSS VERSION

CSS Version	Description
CSS1	Classic CSS implementation that introduced text, list, box, margin, border, color, and background properties. Initially defined in 1996, most every feature of CSS1 is supported in Web browsers, but small quirks do exist around some lesser-used features like `white-space`, `letter-spacing`, `display`, and others. Some problems with CSS1 support are more significant in older, pre–Internet Explorer 7 browsers.
CSS2	Specification that is primarily known for positioning and media, particularly print style sheet features. Many aspects of CSS2, such as aural style sheets, were never widely implemented and were removed in a later iteration of the CSS specification.
CSS 2.1	A revision of the CSS2 specification that makes some corrections and is normalized to more clearly represent what most browser vendors have implemented. Note that many CSS2 features removed from this specification are found in CSS3 modules. This is currently the recommended CSS specification for study and use.
CSS3	Modularized specification of CSS. Various modules extend and improve aspects of previous CSS versions; for example, the CSS3 Color module addresses color correction, transparency, and more, while the CSS3 Fonts module addresses features to add effects to fonts, adjust their display, and even download custom fonts. Some modules are all new, like the Transitions and Animations modules, and others are quite old looking with activity levels suggesting they are abandoned or near abandon. Whatever the situation, when it comes to CSS3, readers are encouraged to check the CSS3 Web site and test support carefully.

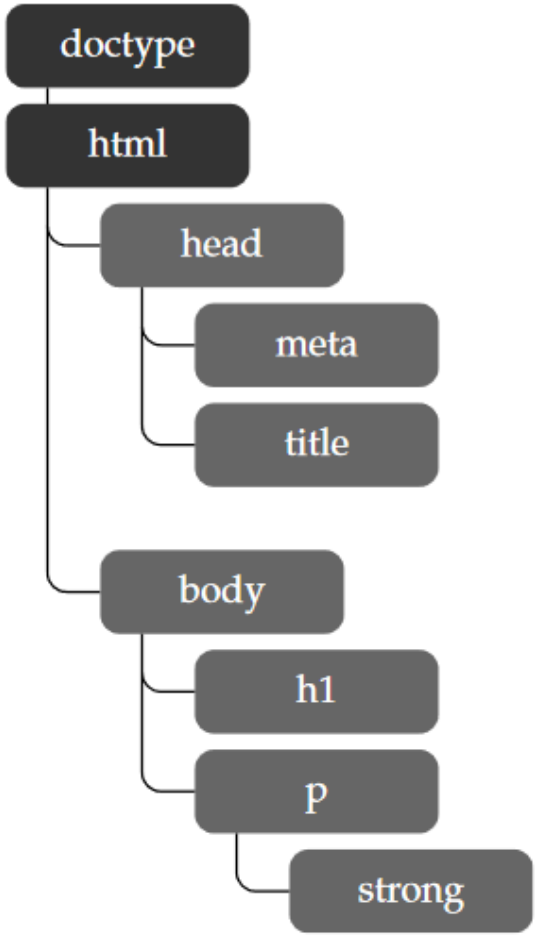
THE SPECIFICATIONS OF CSS

(X)HTML Tags or Attributes	CSS Property Equivalent(s)	Notes
`<center>`	`text-align`, `margin`	Values for `margin` such as `auto` generally are used when centering blocks with `text-align` for content.
`<font>`	`font-family`, `font-size`, `color`	
`align` attributes	`text-align`, `float`	In the case of some elements such as `<img>`, the CSS `float` property is more appropriate than `text-align`.
Color attributes for `<body>`	`color`, `background-color`	To set some of the body attributes like `link`, `vlink`, `alink`, pseudo-classes `:link`, `:visited`, `:active` should be used for `<a>` tags.
Background image attributes for `<body>`, `<table>`, and `<td>`	`background-image`	
The `type` and **`start`** attributes on lists and list items	`list-style-type`, CSS counters	Single CSS properties can't directly substitute some features.
The `clear` attribute for ` `	`clear`	
`<s>`, `<strike>`	`text-decoration: line-through`	
`<u>`	`text-decoration: underline;`	
`<blink>`	`text-decoration: blink`	Not supported in all browsers.

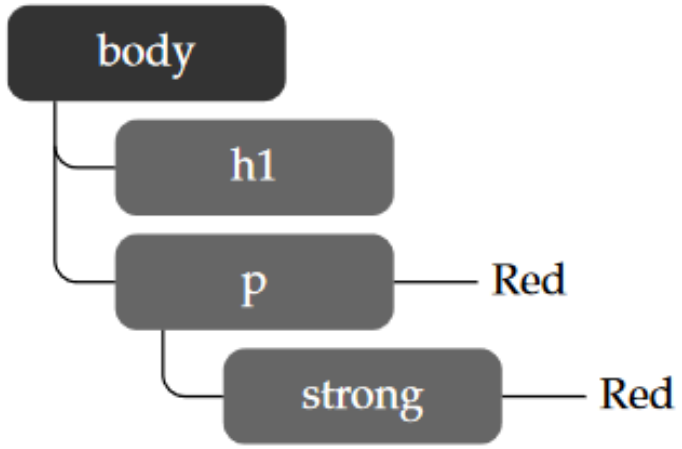
APPLYING STYLE TO A DOCUMENT

	External Style Sheets	Document-Wide Style	Inline Style
Pros	• Can set and update styles for many documents at once. • Style information is cached by the browser, so there's no need to repeat.	• Can easily control style document by document. • No additional network requests to retrieve style information.	• Can easily control style to a single character instance. • Overrides any external or document styles in the absence of `!important` directive.
Cons	• Requires extra download round-trip for the style sheet, which might delay page rendering, particularly when multiple files are in use. • In some cases when `@import` is used, the browser may cause a rendering "flash" under slow loading conditions.	• Need to reapply style information for other documents, bulking up the document and making it more difficult to apply updates.	• Need to reapply style information throughout the document and outside documents. • Bound too closely to markup, making it even more difficult to update than other approaches.
Example	`<link rel="stylesheet" href="main.css" type="text/css" media="screen">` ***NOTE*** *A trailing slash is needed for XHTML.*	`<style type="text/css" media="all">` `h1 {color: red;}` `</style>`	`<h1 style="color: red;">` `I am red!` `</h1>`

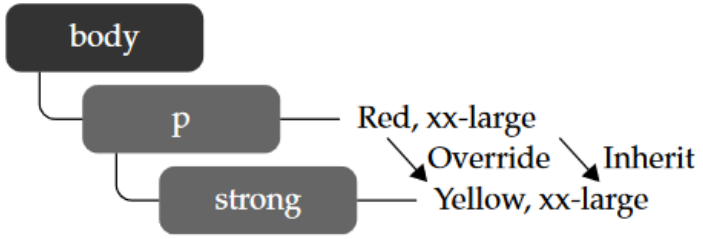
DOCUMENT STRUCTURE AND CSS INHERITANCE



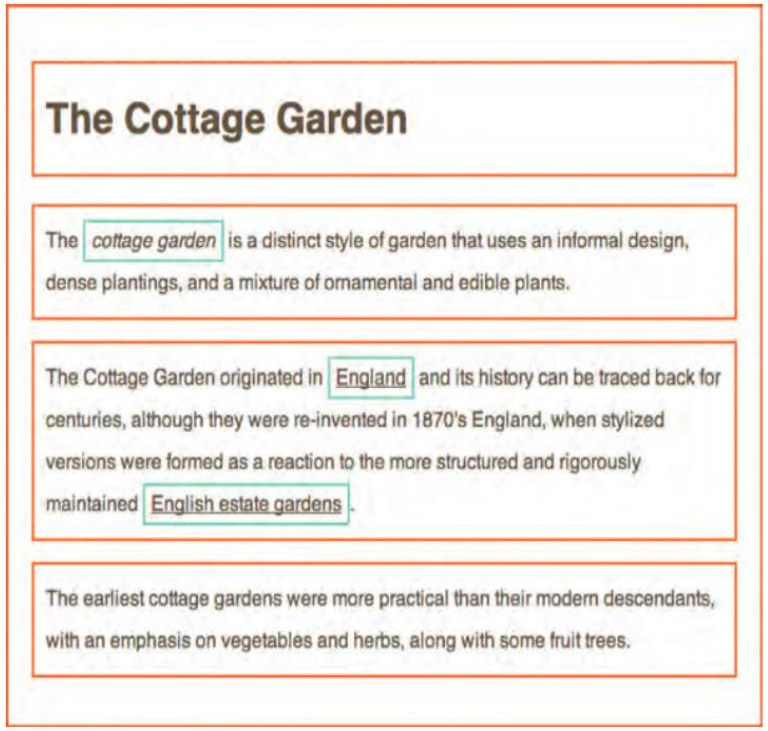
p {color:red;}




```
p {color:red; font-size: xx-large}
strong {color: yellow;}
```



CSS ALLOWS YOU TO CREATE RULES THAT CONTROL THE WAY THAT EACH INDIVIDUAL BOX (AND THE CONTENTS OF THAT BOX) IS PRESENTED.



In this example, block level elements are shown with red borders, and inline elements have green borders.

The `<body>` element creates the first box, then the `<h1>`, `<h2>`, `<p>`, `<i>`, and `<a>` elements each create their own boxes within it.

Using CSS, you could add a border around any of the boxes, specify its width and height, or add a background color. You could also control text inside a box — for example, its color, size, and the typeface used.

EXAMPLE STYLES

BOXES

Width and height
Borders (color, width, and style)
Background color and images
Position in the browser window.

TEXT

Typeface
Size
Color
Italics, bold, uppercase, lowercase, small-caps

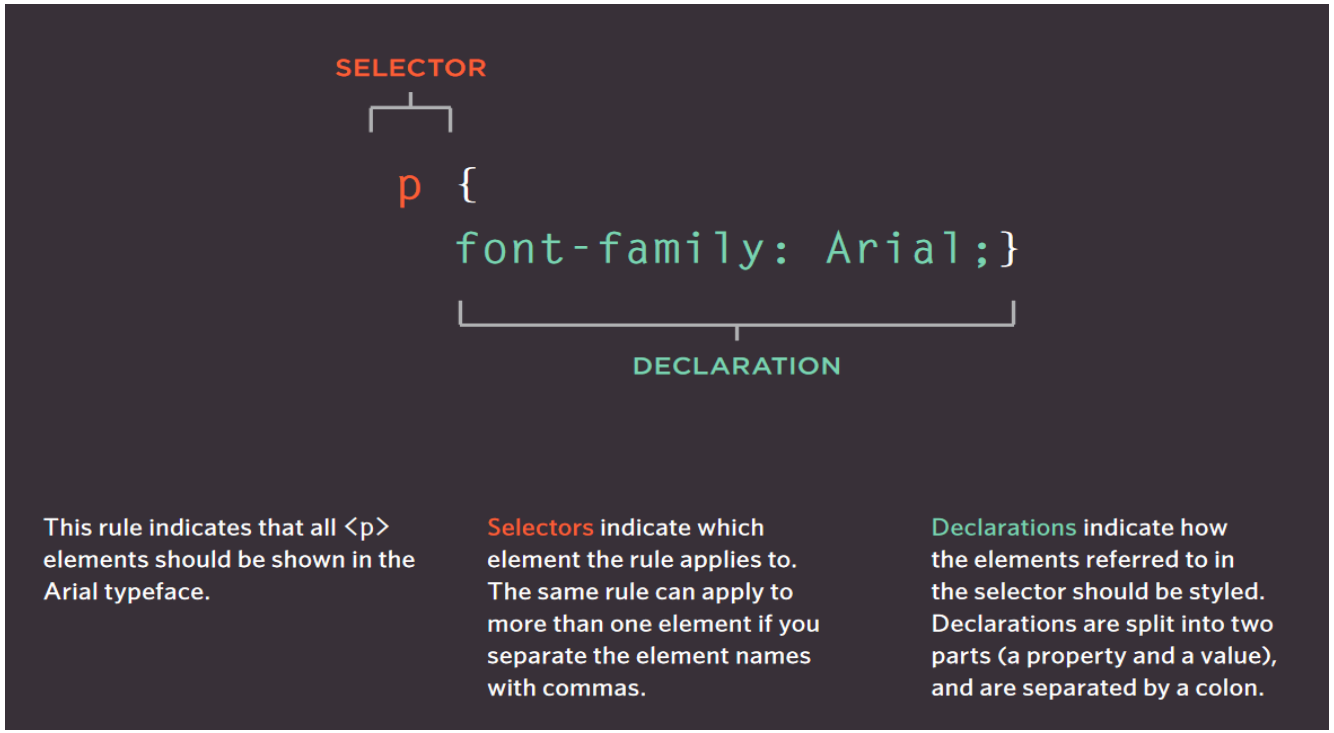
SPECIFIC

There are also specific ways in which you can style certain elements such as lists, tables, and forms.

CSS ASSOCIATES STYLE RULES WITH HTML ELEMENTS

CSS works by associating rules with HTML elements. These rules govern how the content of specified elements should be displayed. A CSS rule contains two parts:

a **selector** and a **declaration.**



CSS PROPERTIES AFFECT HOW ELEMENTS ARE DISPLAYED

CSS declarations sit inside curly brackets and each is made up of two parts: a **property** and a **value**, separated by a colon. You can specify several properties in one declaration, each separated by a semi-colon.

```
h1, h2, h3 {
            font-family: Arial;
            color: yellow;}
```

PROPERTY VALUE

This rule indicates that all <h1>, <h2> and <h3> elements should be shown in the Arial typeface, in a yellow color.

Properties indicate the aspects of the element you want to change. For example, color, font, width, height and border.

Values specify the settings you want to use for the chosen properties. For example, if you want to specify a color property then the value is the color you want the text in these elements to be.

THINGS TO REMEMBER:

<link>

The <link> element can be used in an HTML document to tell the browser where to find the CSS file used to style the page. It is an empty element (meaning it does not need a closing tag), and it lives inside the <head> element. It should use three attributes:

href

This specifies the path to the CSS file (which is often placed in a folder called css or styles).

type

This attribute specifies the type of document being linked to. The value should be text/css.

rel

This specifies the relationship between the HTML page and the file it is linked to. The value should be stylesheet when linking to a CSS file.

An HTML page can use more than one CSS style sheet. To do this it could have a <link> element for every CSS file it uses. For example, some authors use one CSS file to control the presentation (such as fonts and colors) and a second to control the layout.

<style>

You can also include CSS rules within an HTML page by placing them inside a <style> element, which usually sits inside the <head> element of the page.

The <style> element should use the type attribute to indicate that the styles are specified in CSS. The value should be text/css.

When building a site with more than one page, you should use an external CSS style sheet. This:

- Allows all pages to use the same style rules (rather than repeating them in each page).
- Keeps the content separate from how the page looks.
- Means you can change the styles used across all pages by altering just one file (rather than each individual page).

EXAMPLE IN HTML AND CSS

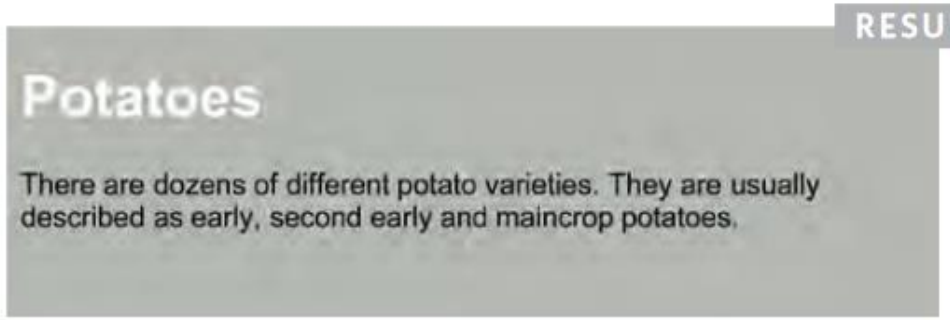
chapter-10/using-external-css.html HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using External CSS</title>
    <link href="css/styles.css" type="text/css"
      rel="stylesheet" />
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

chapter-10/styles.css CSS

```
body {
    font-family: arial;
    background-color: rgb(185,179,175);}
h1 {
    color: rgb(255,255,255);}
```

RESULT

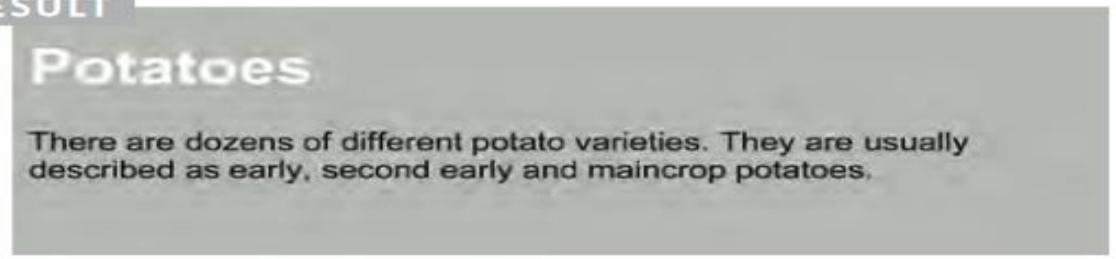


INTERNAL CSS

HTML + CSS chapter-10/using-internal-css.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using Internal CSS</title>
    <style type="text/css">
      body {
          font-family: arial;
          background-color: rgb(185,179,175);}
      h1 {
          color: rgb(255,255,255);}
    </style>
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

RESULT



CSS SELECTORS

- There are many different types of CSS selector that allow you to target rules to specific elements in an HTML document.
- The table on the opposite page introduces the most commonly used CSS selectors.
- CSS selectors are case sensitive, so they must match element names and attribute values exactly.

chapter-10/css-selectors.html HTML

```html
<!DOCTYPE html>
<html>
  <head>
    <title>CSS Selectors</title>
  </head>
  <body>
    <h1 id="top">Kitchen Garden Calendar</h1>
    <p id="introduction">Here you can read our
       handy guide about what to do when.</p>
    <h2>Spring</h2>
    <ul>
      <li><a href="mulch.html">
             Spring mulch vegetable beds</a></li>
      <li><a href="potato.html">
             Plant out early potatoes</a></li>
      <li><a href="tomato.html">
             Sow tomato seeds</a></li>
      <li><a href="beet.html">
             Sow beet seeds</a></li>
      <li><a href="zucchini.html">
             Sow zucchini seeds</a></li>
      <li><a href="rhubarb.html">
             Deadhead rhubarb flowers</a></li>
    </ul>
    <p class="note">
       This page was written by
      <a href="mailto:ivy@example.org">
         ivy@example.org</a> for
      <a href="http://www.example.org">Example</a>.
    </p>
    <p>
      <a href="#top">Top of page</a>
    </p>
  </body>
</html>
```

CSS SELECTORS

SELECTOR	MEANING	EXAMPLE
UNIVERSAL SELECTOR	Applies to all elements in the document	`* {}` Targets all elements on the page
TYPE SELECTOR	Matches element names	`h1, h2, h3 {}` Targets the <h1>, <h2> and <h3> elements
CLASS SELECTOR	Matches an element whose `class` attribute has a value that matches the one specified after the period (or full stop) symbol	`.note {}` Targets any element whose `class` attribute has a value of `note` `p.note {}` Targets only <p> elements whose `class` attribute has a value of `note`
ID SELECTOR	Matches an element whose `id` attribute has a value that matches the one specified after the pound or hash symbol	`#introduction {}` Targets the element whose `id` attribute has a value of `introduction`
CHILD SELECTOR	Matches an element that is a direct child of another	`li>a {}` Targets any <a> elements that are children of an element (but not other <a> elements in the page)
DESCENDANT SELECTOR	Matches an element that is a descendent of another specified element (not just a direct child of that element)	`p a {}` Targets any <a> elements that sit inside a <p> element, even if there are other elements nested between them
ADJACENT SIBLING SELECTOR	Matches an element that is the next sibling of another	`h1+p {}` Targets the first <p> element after any <h1> element (but not other <p> elements)
GENERAL SIBLING SELECTOR	Matches an element that is a sibling of another, although it does not have to be the directly preceding element	`h1~p {}` If you had two <p> elements that are siblings of an <h1> element, this rule would apply to both

This presentation is created by: Dr. Pau Abesamis

BASIC SELECTOR

Selector	Description
`*`	Universal selector (all elements)
`div`	Tag selector (all `<div>` elements)
`.blue`	Class selector (all elements with class `blue`)
`.blue.red`	All elements with class `blue` **and** `red` (a type of Compound selector)
`#headline`	ID selector (the element with "id" attribute set to `headline`)
`:pseudo-class`	All elements with pseudo-class
`::pseudo-element`	Element that matches pseudo-element
`:lang(en)`	Element that matches :lang declaration, for example `<span lang="en">`
`div > p`	child selector

HOW CSS RULES CASCADE?

- If there are two or more rules that apply to the same element, it is important to understand which will take precedence.

1) **Last Rule -** If the two selectors are identical, the latter of the two will take precedence. Here you can see the second i selector takes precedence over the first.

2) **Specificity** - If one selector is more specific than the others, the more specific rule will take precedence over more general ones. In this example:

    h1 is more specific than *

    p b is more specific than p

    p#intro is more specific than p

chapter-10/cascade.html HTML

```html
<h1>Potatoes</h1>
<p id="intro">There are <i>dozens</i> of different
   <b>potato</b> varieties.</p>
<p>They are usually described as early, second early
   and maincrop potatoes.</p>
```

CSS

```css
* {
  font-family: Arial, Verdana, sans-serif;}
h1 {
  font-family: "Courier New", monospace;}
i {
  color: green;}
i {
  color: red;}
b {
  color: pink;}
p b {
  color: blue !important;}
p b {
  color: violet;}
p#intro {
  font-size: 100%;}
p {
  font-size: 75%;}
```

RESULT

Potatoes

There are *dozens* of different potato varieties.

They are usually described as early, second early and maincrop potatoes.

INHERITANCE

- If you specify the font-family or color properties on the <body> element, they will apply to most child elements. This is because the value of the font-family property is inherited by child elements. It saves you from having to apply these properties to as many elements (and results in simpler style sheets).
- You can force a lot of properties to inherit values from their parent elements by using inherit for the value of the properties. In this example, the element with a class called page inherits the padding size from the CSS rule that applies to the <body> element

INHERITANCE

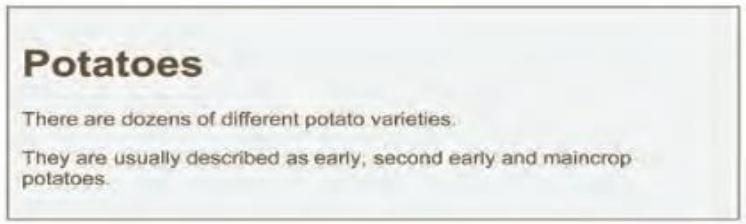
HTML chapter-10/inheritance.html

```
<div class="page">
  <h1>Potatoes</h1>
  <p>There are dozens of different potato
     varieties.</p>
  <p>They are usually described as early, second
     early and maincrop potatoes.</p>
</div>
```

CSS

```
body {
  font-family: Arial, Verdana, sans-serif;
  color: #665544;
  padding: 10px;}
.page {
  border: 1px solid #665544;
  background-color: #efefef;
  padding: inherit;}
```

RESULT



WHY USE EXTERNAL STYLE SHEETS?

- When building a website there are several advantages to placing your CSS rules in a separate style sheet.

All of your web pages can share the same style sheet. This is achieved by using the `<link>` element on each HTML page of your site to link to the same CSS document. This means that the same code does not need to be repeated in every page (which results in less code and smaller HTML pages).

Therefore, once the user has downloaded the CSS stylesheet, the rest of the site will load faster. If you want to make a change to how your site appears, you only need to edit the one CSS file and all of your pages will be updated. For example, you can change the style of every `<h1>` element by altering the one CSS style sheet, rather than changing the CSS rules on every page. The HTML code will be easier to read and edit because it does not have lots of CSS rules in the same document. It is generally considered good practice to have the content of the site separated from the rules that determine how it appears.

SOMETIMES YOU MIGHT CONSIDER PLACING CSS RULES IN THE SAME PAGE AS YOUR HTML CODE

If you are just creating a single page, you might decide to put the rules in the same file to keep everything in one place. (However, many authors would consider it better practice to keep the CSS in a separate file.)

If you have one page which requires a few extra rules (that are not used by the rest of the site), you might consider using CSS in the same page. (Again, most authors consider it better practice to keep all CSS rules in a separate file.)

Most of the examples in this book place the CSS rules in the `<head>` of the document (using the `<style>` element) rather than a separate document. This is simply to save you opening two files to see how the CSS examples work.

DIFFERENT VERSIONS OF CSS & BROWSERS QUIRKS

- CSS1 was released in 1996 and CSS2 followed two years later. Work on CSS3 has been ongoing but the major browsers have already started to implement it.

In the same way that there have been several versions of HTML, there have also been different versions of CSS.

Browsers did not implement all CSS features at once, so some older browsers do not support every property.

This is mentioned when it is likely to affect you, along with notes where CSS properties might not behave as expected.

BACKGROUND COLOR

The `background-color` property sets the background color of an element using a color value or through keywords, such as `transparent`, `inherit` or `initial`.

- **transparent**, specifies that the background color should be transparent. This is default.
- **inherit**, inherits this property from its parent element.
- **initial**, sets this property to its default value.

This can be applied to all elements, and `::first-letter`/`::first-line` pseudo-elements.

Colors in CSS can be specified by different methods.

Color names

CSS

```
div {
  background-color: red;  /* red */
}
```

HTML

```
<div>This will have a red background</div>
```

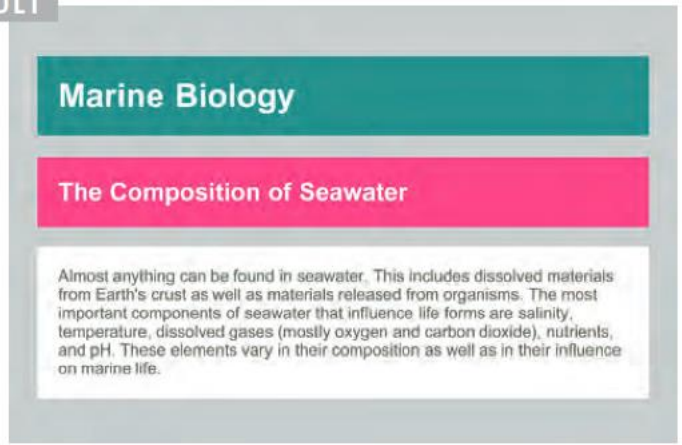
- The example used above is one of several ways that CSS has to represent a single color.

CSS BACKGROUND-COLOR

CSS chapter-11/background-color.html

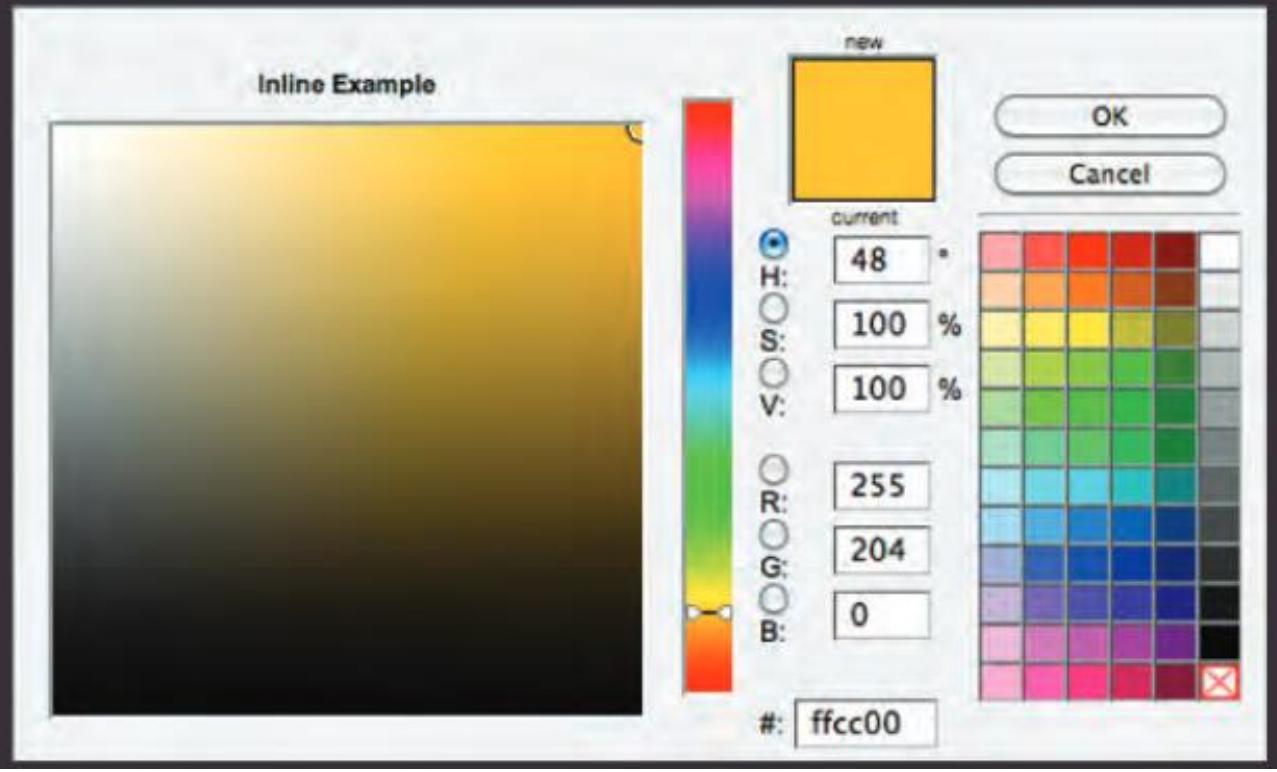
```
body {
  background-color: rgb(200,200,200);}
h1 {
  background-color: DarkCyan;}
h2 {
  background-color: #ee3e80;}
p {
  background-color: white;}
```

RESULT



- By default, most browser windows have a white background, but browser users can set a background color for their windows, so if you want to be sure that the background is white you can use the background-color property on the <body> element

COLOR PICKER



Color picking tools are available in image editing programs like Photoshop and GIMP. You can see the RGB values specified next to the radio buttons that say R, G, B.

The hex value is provided next to the pound or hash # symbol. There is also a good color picking tool at: `colorschemedesigner.com`

UNDERSTANDING COLORS

RGB VALUES

Values for red, green, and blue are expressed as numbers between 0 and 255.

`rgb(102,205,170)`

This color is made up of the following values:
102 red
205 green
170 blue

HEX CODES

Hex values represent values for red, green, and blue in hexadecimal code.

`#66cdaa`

The value of the red, 102, is expressed as 66 in hexadecimal code. The 205 of the green is expressed as `cd` and the 170 of the blue equates to `aa`.

COLOR NAMES

Colors are represented by predefined names. However, they are very limited in number.

`MediumAquaMarine`

There are 147 color names supported by browsers (this color is `MediumAquaMarine`). Most consider this to be a limited color palette, and it is hard to remember the name for each of the colors so (apart from white and black) they are not commonly used.

HUE

Hue is near to the colloquial idea of color. Technically speaking however, a color can also have saturation and brightness as well as hue.

SATURATION

Saturation refers to the amount of gray in a color. At maximum saturation, there would be no gray in the color. At minimum saturation, the color would be mostly gray.

BRIGHTNESS

Brightness (or "value") refers to how much black is in a color. At maximum brightness, there would be no black in the color. At minimum brightness, the color would be very dark.

CONTRAST

When picking foreground and background colors, it is important to ensure that there is enough contrast for the text to be legible.

LOW CONTRAST

Text is harder to read when there is low contrast between background and foreground colors.

A lack of contrast is particularly a problem for those with visual impairments and color blindness.

It also affects those with poor monitors and sunlight on their screens (which is increasingly common as people use handheld devices outdoors).

HIGH CONTRAST

Text is easier to read when there is higher contrast between background and foreground colors.

If you want people to read a lot of text on your page, however, then too much contrast can make it harder to read, too.

If text is reversed out (a light color on a dark background), you can increase the height between lines and the weight of the font to make it easier to read.

MEDIUM CONTRAST

For long spans of text, reducing the contrast a little bit improves readability.

You can reduce contrast by using dark gray text on a white background or an off-white text on a dark background.

ATTRIBUTE SELECTOR

SELECTOR	MEANING	EXAMPLE
EXISTENCE	`[]` Matches a specific attribute (whatever its value)	`p[class]` Targets any <p> element with an attribute called `class`
EQUALITY	`[=]` Matches a specific attribute with a specific value	`p[class="dog"]` Targets any <p> element with an attribute called `class` whose value is `dog`
SPACE	`[~=]` Matches a specific attribute whose value appears in a space-separated list of words	`p[class~="dog"]` Targets any <p> element with an attribute called `class` whose value is a list of space-separated words, one of which is `dog`
PREFIX	`[^=]` Matches a specific attribute whose value begins with a specific string	`p[attr^"d"]` Targets any <p> element with an attribute whose value begins with the letter "d"
SUBSTRING	`[*=]` Matches a specific attribute whose value contains a specific substring	`p[attr*"do"]` Targets any <p> element with an attribute whose value contains the letters "do"
SUFFIX	`[$=]` Matches a specific attribute whose value ends with a specific string	`p[attr$"g"]` Targets any <p> element with an attribute whose value ends with the letter "g"

COMMENTS

Single Line

```css
/* This is a CSS comment */
div {
   color: red; /* This is a CSS comment */
}
```

Multiple Line

```css
/*
    This
    is
    a
    CSS
    comment
*/
div {
   color: red;
}
```

INTERNAL STYLES

```
<head>
    <style>
        h1 {
            color: green;
            text-decoration: underline;
        }
        p {
            font-size: 25px;
            font-family: 'Trebuchet MS', sans-serif;
        }
    </style>
</head>
<body>
    <h1>Hello world!</h1>
    <p>I ♥ CSS</p>
</body>
```

Hello world!

I ♥ CSS

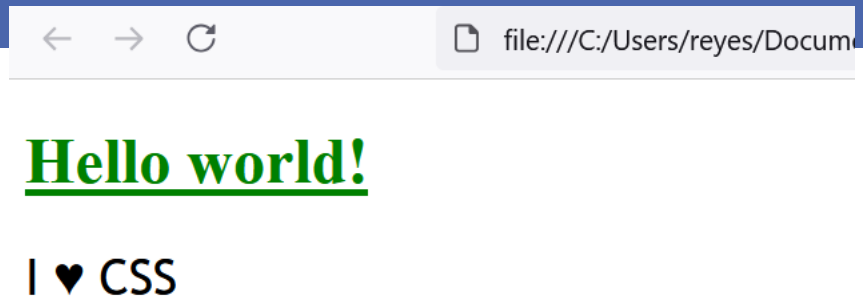
I) EXTERNAL STYLESHEET

hello-world.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        <h1>Hello world!</h1>
        <p>I ♥ CSS</p>
    </body>
</html>
```

style.css

```
h1 {
    color: green;
    text-decoration: underline;
}
p {
    font-size: 25px;
    font-family: 'Trebuchet MS', sans-serif;
}
```



CSS SEATWORK#1

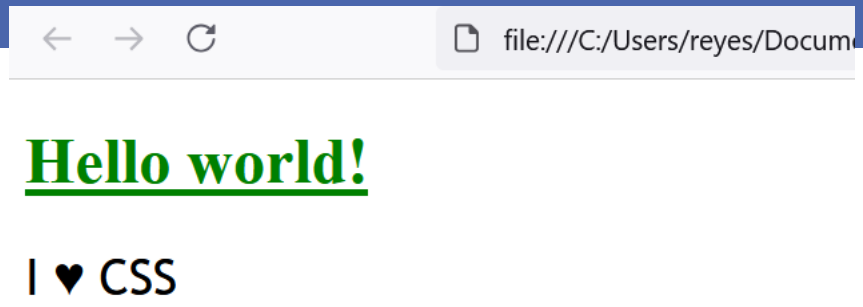
I) EXTERNAL STYLESHEET

hello-world.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        <h1>Hello world!</h1>
        <p>I ♥ CSS</p>
    </body>
</html>
```

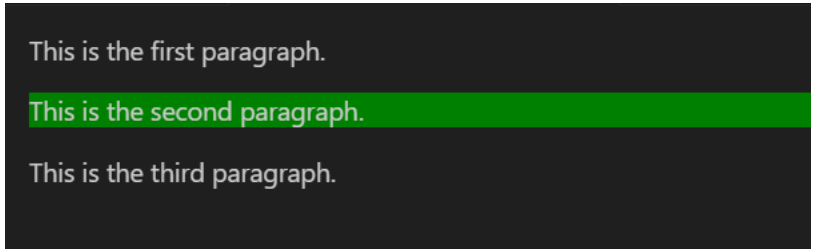
style.css

```
h1 {
    color: green;
    text-decoration: underline;
}
p {
    font-size: 25px;
    font-family: 'Trebuchet MS', sans-serif;
}
```



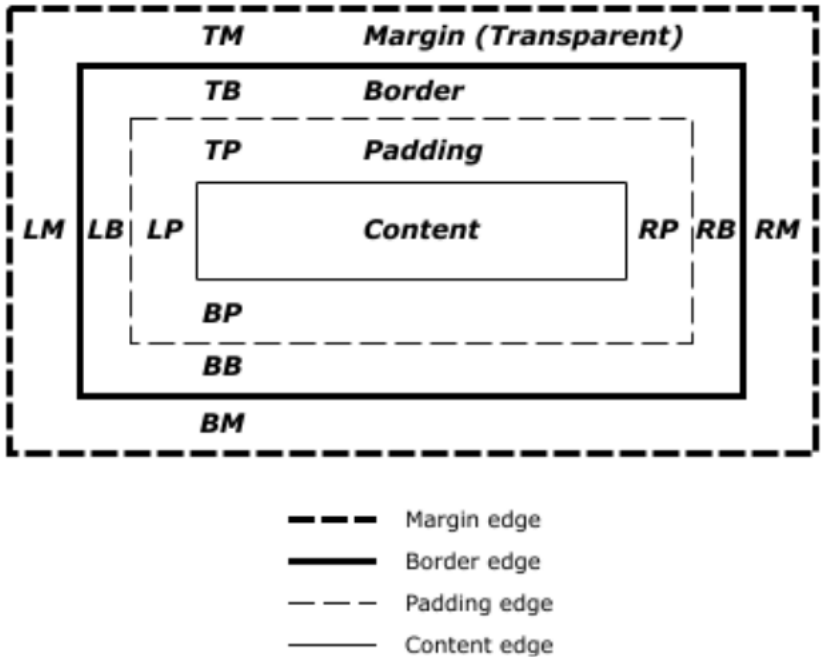
2. CREATE A HTML WITH INLINE CSS

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Id Selector Example</title>
<style type="text/css" media="all">
 #p2 {background-color: green;}
</style>
</head>
<body>
<p>This is the first paragraph.</p>
<p id="p2">This is the second paragraph.</p>
<p>This is the third paragraph.</p>
</body>
</html>
```



CSS BOX MODEL

- The browser creates a rectangle for each element in the HTML document. The Box Model describes how the padding, border, and margin are added to the content to create this rectangle.



CSS BOX MODEL

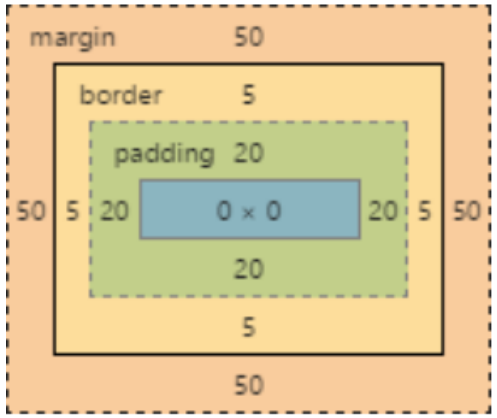
The perimeter of each of the four areas is called an edge. Each edge defines a box.

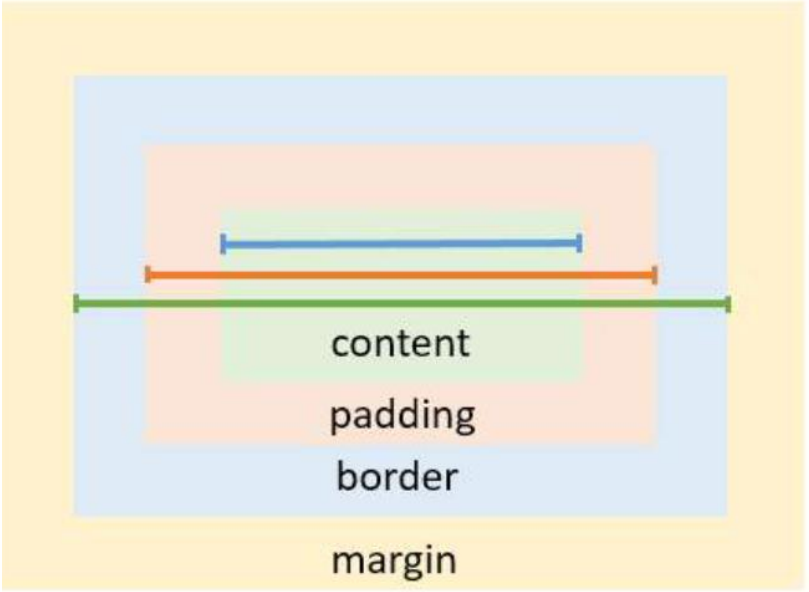
- The innermost rectangle is the **content box**. The width and height of this depends on the element's rendered content (text, images and any child elements it may have).
- Next is the padding box, as defined by the **padding property**. If there is no padding width defined, the padding edge is equal to the content edge.
- Then we have the border box, as defined by the **border property**. If there is no border width defined, the border edge is equal to the padding edge.
- The outermost rectangle is the **margin box**, as defined by the margin property. If there is no margin width defined, the margin edge is equal to the border edge.

EXAMPLE OF CSS BOX MODEL

div { border: 5px solid red; margin: 50px; padding: 20px; }

This CSS styles all `div` elements to have a top, right, bottom and left border of `5px` in width; a top, right, bottom and left margin of `50px`; and a top, right, bottom, and left padding of `20px`. Ignoring content, our generated box will look like this:





width

- **content-box**: content
- **padding-box**: content + padding
- **border-box**: content + padding + border

REFERENCES

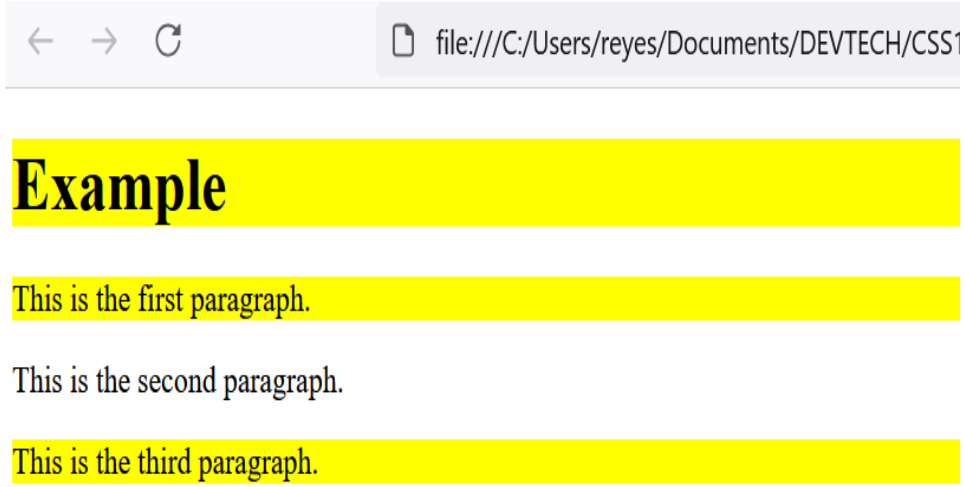
- https://books.goalkicker.com/CSSBook/CSSNotesForProfessionals.pdf
- https://www.w3.org/TR/CSS22/box.html#mpb-examples
- https://wtf.tw/ref/duckett.pdf
- https://www.dcpehvpm.org/E-Content/BCA/BCA-II/Web%20Technology/the-complete-reference-html-css-fifth-edition.pdf
- https://github.com/TheOdinProject/css-exercises/tree/main/advanced-html-css/animation/01-button-hover

CSS SEATWORK#2

3. CSS WITH CLASSRULES

- The class attribute is used to define the name(s) of the class(es) to which a particular tag belongs. Unlike id values, class values don't have to be unique because many elements can be members of the same class.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Class Selector Example</title>
<style type="text/css" media="all">
  .veryimportant  {background-color: yellow;}
</style>
</head>
<body>
<h1 class="veryimportant">Example</h1>
<p class="veryimportant">This is the first paragraph.</p>
<p>This is the second paragraph.</p>
<p class="veryimportant">This is the third paragraph.</p>
</body>
</html>
```

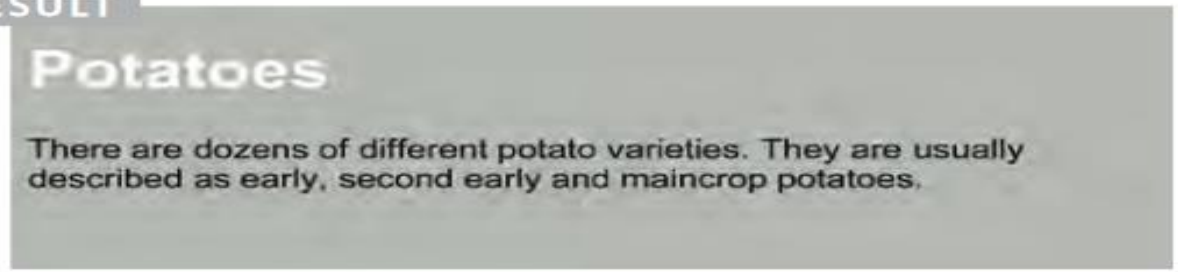


4. INTERNAL CSS

HTML + CSS chapter-10/using-internal-css.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using Internal CSS</title>
    <style type="text/css">
      body {
          font-family: arial;
          background-color: rgb(185,179,175);}
      h1 {
          color: rgb(255,255,255);}
    </style>
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

RESULT



GROUP REPORTING #1

REPORTING ACTIVITY (BY PAIR)

1) Chapter 8 – Margins & Chapter 10 – Boarder
2) Chapter 11 – Outline & Chapter 12 - Overflow
3) Chapter 13 – Media Queries & Chapter 25 - Table
4) Chapter 15 – Typography & Chapter 26 - Transition
5) Chapter 17 – Cascading and Specificity & Chapter 18 – Colors
6) Chapter 22 – Positioning, Chapter 23 – Layout Control and Chapter 24 - GRID

INSTRUCTIONS: CSS PRESENTATION

- Download the book: CSSNotesforProfessionals
  https://books.goalkicker.com/CSSBook/CSSNotesForProfessionals.pdf
- Presentation Content:
  - Define the topic (use the book and other resource)
  - Provide Example/Hands-on
  - Create 1 exercise for the class
- Maximum of 15 mins presentation
- Prepare for Q&A
- The presentation should be uploaded in GITHUB.