# CSS Color Properties

## 1. Background and Text Colors

* **background-color**\*\* Property:\*\*
  + Sets the background color of an element.
  + **Syntax:**

element {

background-color: red;

}

* + **Example:**  
    Setting the entire webpage's background:

body {

background-color: red;

}

* **color**\*\* Property:\*\*
  + Sets the text color of an element.
  + **Syntax:**
  + element {
  + color: blue;
  + }
  + **Example:**  
    Changing the text color of a heading:
  + h1 {
  + color: blue;
  + }

## 2. Using Named Colors

* **Definition:**  
  Named colors are predefined color names recognized by CSS (e.g., red, blue, whitesmoke).
* **Examples of Named Colors:**
  + cornflowerblue
  + cadetblue
  + dimgrey
  + olivedrab
* **Reference:**  
  Use MDN Docs for the complete list and details on each named color.

## 3. Hex Codes for Colors

* **What are Hex Codes?**
  + Hex codes represent colors using a six-digit combination of numbers and letters (e.g., #5d3891).
  + They correspond to RGB (Red, Green, Blue) values, where each component is given on a scale from 0 to 255.
* **Example:**  
  Given an RGB value of (93, 56, 145):
  + Convert each to hexadecimal:
    - 93 → 5d
    - 56 → 38
    - 145 → 91
  + Combined hex code: #5d3891
  + **Usage in CSS:**
  + h2 {
  + color: #5d3891;
  + }

## 4. Leveraging Color Palettes

* **When to Use:**  
  Use custom color palettes when the default named colors do not meet your design needs.
* **Recommended Tool:**  
  [Color Hunt](https://colorhunt.co/)
  + Offers professionally designed color palettes.
  + Simply copy the hex code provided by Color Hunt and use it in your CSS.

# . CSS Font Properties

## 1. Overview

* **Purpose:**  
  Change the appearance of text on websites.
* **Key Properties Covered:**
  + **font-size**
  + **font-weight**
  + **font-family**
  + **text-align**

## 2. Font Size

### 2.1 Units of Measurement

* **Pixels (px):**
  + **Definition:** 1px ≈ 1/96th of an inch (≈0.26mm).
  + **Usage Example:**
  + p {
  + font-size: 20px;

}

* **Points (pt):**
  + **Definition:** 1pt ≈ 1/72nd of an inch (≈0.35mm).
  + **Usage Example:**
  + p {
  + font-size: 12pt;

}

* **Relative Units: em & rem**
  + **em:**
    - Relative to the **font-size of its parent element**.
    - **Example:**  
      If the parent is 20px:
    - h1 {
    - font-size: 2em; /\* 2 \* 20px = 40px \*/

}

* + **rem:**
    - Relative to the **root element's (html) font-size**.
    - More consistent for nested elements.
    - **Example:**  
      If the root font-size is 20px:
    - h2 {
    - font-size: 2rem; /\* 2 \* 20px = 40px \*/

}

* **Named Sizes:**
  + Examples: x-large, large, etc.

## 3. Font Weight

* **Definition:**  
  Controls how thick or thin text appears.
* **Values:**
  + **Keywords:** normal, bold
  + **Numeric Scale:** 100 (lightest) to 900 (boldest)
* **Usage Examples:**
* p {
* font-weight: normal; /\* or bold \*/
* }
* h1 {
* font-weight: 900;

}

## 4. Font Family

### 4.1 Choosing Fonts

* **Custom Font Inclusion:**
  + Use external resources like [Google Fonts](https://fonts.google.com).
  + Include the font using a <link> in the HTML <head>:

<link href="https://fonts.googleapis.com/css2?family=Caveat&display=swap" rel="stylesheet">

* **CSS Declaration with Fallbacks:**
  + **Example:**
  + h1 {
  + font-family: 'Caveat', cursive;

}

* + - **Note:**  
      If the custom font fails to load, the browser uses the generic cursive family.

### 4.2 Generic Font Families

* **Common Types:**
  + **Sans-serif:** e.g., Helvetica, Arial
  + **Serif:** e.g., Times New Roman
  + **Cursive:** Curly, artistic fonts
  + **Monospace:** Fixed-width (like typewriter fonts)
  + **Fantasy:** Decorative fonts
* **Handling Multi-word Font Names:**
  + Always wrap in quotation marks.
  + p {
  + font-family: "Times New Roman", serif;

}

## 5. Text Alignment

* **Property:** text-align
* **Values:**
  + left, right, center
  + **Start/End:**
    - Useful for languages that read right-to-left (e.g., Arabic)
* **Usage Example:**
* p {
* text-align: right;

}

## 6. Practical Exercise Overview

* **Task:**  
  Modify a sample HTML file (e.g., font-family.html or index.html) to practice these properties.
* **Steps:**
  1. **Change Color:**
     + Set an element's text to a named color (e.g., coral).
  2. **Adjust Font Size:**
     + Use rem to set an element’s font-size relative to the root (e.g., 2rem).
  3. **Modify Font Weight:**
     + Change font-weight using a numeric value (e.g., 900).
  4. **Set Font Family:**
     + Include a Google Font and specify it with a fallback.
  5. **Align Text:**
     + Adjust text alignment (e.g., text-align: right).
  6. **Experiment with Root Font Size:**
     + Change the root <html> font-size to see how rem scales change accordingly.
* **Observation:**  
  Adjusting the root font size will affect all elements defined with rem units, demonstrating the power of relative sizing.

# CSS Inspection & Chrome Developer Tools

## 1. Introduction

* **Purpose:**  
  Diagnose CSS issues, understand style application, and experiment with live changes.
* **Requirement:**  
  Use the Chrome browser for full access to Developer Tools.

## 2. Opening Developer Tools

* **Via Menu:**  
  Click the three dots → More Tools → Developer Tools.
* **Keyboard Shortcuts:**
  + **Mac:** Command + Option + I
  + **Windows:** Ctrl + Shift + I
  + **F12 Key:** Often works as a shortcut.
* **Context Menu:**  
  Right-click an element and choose **Inspect** to highlight its HTML in the Developer Tools.

## 3. Exploring the Elements Tab

* **Elements Panel:**  
  Shows HTML structure and applied CSS.
* **Styles Section:**  
  Lists CSS rules applied to the selected element, including:
  + **Author Styles:** From your custom CSS files (e.g., styles.css).
  + **User Agent Styles:** Default browser styles (often crossed-out when overridden).

**Example:**  
Inspecting a button might reveal:

button {

background-color: #3498db;

color: white;

}

* **Computed Tab:**  
  Displays the final, computed style values (e.g., actual RGB values for colors).

## 4. Live Editing CSS

* **Modifying Styles:**
  + Click the **+** button in the Styles pane to add new CSS rules.
  + Directly edit property values (e.g., change color: white; to color: black;).
* **Temporary Changes:**
  + Modifications are local and do **not** change the original source files.
  + Refreshing the page reverts changes.

## 5. Using the CSS Overview Feature

* **Access:**  
  Click the three-dot menu within Developer Tools → More Tools → CSS Overview.
* **Functionality:**  
  Provides a summary of:
  + Background colors used
  + Text colors in effect
  + Fonts and their usage
* **Benefit:**  
  Useful for quickly identifying design elements (e.g., what exact green or font family is used).

## 6. Practical Exercise

* **Challenge Site:**  
  Visit [appbrewery.github.io/css-inspection/](https://appbrewery.github.io/css-inspection/) to practice inspecting elements.
* **Quiz Questions (Examples):**
  1. **Named Color of the Body:**
     + Inspect the <body> element.
     + **Answer:** "aliceblue"
  2. **Font Size of the h1:**
     + Inspect the <h1> element.
     + **Answer:** "3rem"
  3. **Font Weight of the h2:**
     + Inspect the <h2> element.
     + **Answer:** "500"
  4. **Font Family of the Paragraph:**
     + Use the **Computed** tab if necessary.
     + **Answer:** "Arial, sans-serif"

## 7. Additional Tips

* **Element Selection:**  
  Use the visual selector tool (the cursor icon) to click on any element for quick inspection.
* **Backup Your Files:**  
  Remember, changes in DevTools are temporary and only affect the live view.
* **Experiment:**  
  Try enabling/disabling individual rules to see which ones are in effect.

# CSS Box Model

## 1. The Box Model Overview

* **Concept:**  
  Every HTML element is treated as a box. Although you cannot see all parts of the box, you can control:
  + **Content Area:** The space where content (like text or images) is displayed.
  + **Padding:** The space between the content and the border.
  + **Border:** The edge around the padding and content.
  + **Margin:** The space outside the border, separating the element from others.
* **Visualizing the Model:**  
  Use the CSS Inspector (found in Chrome Developer Tools) to view the box model of any element. It displays:
  + Width and height (content dimensions)
  + Padding, border, and margin values

## 2. Width and Height

* **Width & Height Properties:**
  + **Definition:**  
    Set the dimensions of an element’s content area.
  + **Units:**  
    Can be defined in pixels (px) or percentages (%).
  + **Example:**

css

CopyEdit

.element {

width: 200px; /\* Fixed width \*/

height: 300px; /\* Fixed height \*/

}

* + **Note:**  
    Changing width or height affects how elements are laid out on the page.

## 3. Borders

* **Border Property Basics:**
  + **Shorthand Property:**  
    Combines three aspects: border thickness, style, and color.
  + **Syntax:**

css

CopyEdit

.element {

border: 10px solid red;

}

* + **Components:**
    - **Thickness:** e.g., 10px
    - **Style:** e.g., solid, dashed
    - **Color:** e.g., red or a hex code like #ff0000
  + **Direction-Specific Borders:**  
    You can override any side after the shorthand is set.

css

CopyEdit

.element {

border: 30px solid black;

border-top-width: 0px; /\* Removes the top border \*/

}

* **Border-Width with Multiple Values:**
  + **Four Values (Clockwise Order):**  
    top right bottom left
  + **Two Values:**  
    First value applies to top & bottom, the second to left & right.
  + **Example:**

css

CopyEdit

.element {

border-width: 0px 20px; /\* Top & Bottom = 0, Left & Right = 20px \*/

}

* **Important Note:**  
  The border is drawn **outside** the content area without changing the defined width and height of the element’s content box.

## 4. Padding

* **Padding Definition:**
  + **Purpose:**  
    Creates space between the content and the border.
  + **Usage:**  
    It does not affect the element’s defined width/height but pushes the border outward.
  + **Example:**

.paragraph {

padding: 20px; /\* 20px space on all sides \*/

}

* **Multiple Values:**  
  Works similar to margin and border:
  + One value: all sides
  + Two values: first for top & bottom, second for left & right
  + Four values: top, right, bottom, left (clockwise order)

## 5. Margin

* **Margin Definition:**
  + **Purpose:**  
    Creates space **outside** the border, separating the element from other elements.
  + **Example:**

css

CopyEdit

.element {

margin: 10px; /\* 10px on all sides \*/

}

* **Spacing Between Elements:**  
  When two elements each have a margin (e.g., 10px on the bottom of one and 10px on the top of the next), the space between them becomes 20px.
* **Multiple Values:**  
  Like padding and border, you can specify:
  + One value: all sides
  + Two values: vertical and horizontal
  + Four values: top, right, bottom, left

## 

## 6. Creating Custom Layouts with the Box Model

* **Grouping Content with Divs:**
  + **Div Element:**  
    An HTML container used to group multiple elements.
  + **Purpose:**  
    To apply common CSS styles or layout properties to a group of elements.
  + **Example:**

html

CopyEdit

<div class="content-box">

<img src="bacon.jpg" alt="I Love Bacon">

<p>Delicious bacon description.</p>

</div>

* + **Usage:**  
    Style the container (div) to control spacing, alignment, and grouping.
* **Example Calculation:**
  + **Scenario:**  
    A div with a width of 200px, 20px padding on each side, and a 10px border on each side.
  + **Calculation:**  
    Total horizontal space = 200px (content) + 20px (left padding) + 20px (right padding) + 10px (left border) + 10px (right border) = 260px.
  + **Usage:**  
    Use margin properties to correctly position adjacent boxes.

## 7. Debugging with Chrome Developer Tools & Extensions

* **Chrome Developer Tools:**
  + **Inspect Elements:**  
    Right-click any element and select **Inspect** to view its box model and applied styles.
  + **Live Editing:**  
    Modify CSS live in the Styles pane; changes are temporary.
  + **Computed Tab:**  
    See the final computed styles, which are helpful to understand how properties are applied.
* **CSS Inspector – Box Model View:**  
  Shows margin, border, padding, and content size. You can adjust these values to see real-time changes.
* **Using Pesticide Extension:**
  + **Purpose:**  
    A Chrome extension that outlines every element (div, section, etc.) with a visible border.
  + **Benefit:**  
    Helps to visualize and debug the layout by showing the boundaries of each element.
  + **How to Use:**  
    Install from the Chrome Web Store, activate it, and see the boxes highlighted on your webpage.

## 8. Practical Exercise Overview

* **Task:**  
  A challenge where you edit an HTML file (e.g., index.html) containing several TODOs.
* **Requirements Include:**
  + Setting fixed widths and heights (e.g., 200px for divs).
  + Adding padding to elements (e.g., 20px for paragraph elements).
  + Creating borders with different widths on different sides:
    - Example: 20px border on top and bottom, 10px on left and right.
  + Using margin properties to align divs precisely (touching corner-to-corner).
  + Adjusting individual elements using unique IDs for specific styles.
* **Example Walkthrough:**
  + **Div with Padding & Border:**

css

CopyEdit

#div1 {

width: 200px;

height: 200px;

padding: 20px;

border: 10px solid black;

}

* + **Overriding Border Widths:**

css

CopyEdit

#div2 {

border-width: 20px 10px; /\* 20px top/bottom, 10px left/right \*/

}

* + **Removing Default Margin from Paragraph:**

css

CopyEdit

p {

margin: 0;

}

* + **Positioning Adjacent Boxes:**  
    Calculate the total outer width (e.g., 200px content + padding + border = 260px) and apply a corresponding margin-left to align the next div.

## Final Thoughts

* **Key Takeaways:**
  + Understand how the content area, padding, border, and margin form the box model.
  + Use precise values and multiple-value syntaxes (one, two, or four values) to control each side of an element.
  + Utilize Chrome Developer Tools and extensions like Pesticide to debug and refine your layouts.
  + Practice by following real-world exercises that simulate client instructions, ensuring boxes align exactly as specified.

These notes should provide you with a complete reference for the discussed CSS properties and how they work together within the box model. Use these concepts to build precise and well-structured web layouts.