158.258 Web Development

Cascading Style Sheets (CSS) and HTML

School of Mathematical & Computational Sciences
College of Sciences, Massey University
(AKLI, DISD & MTUI)

2022-08-03

Section 1: Learning Objectives

- After studying this topic, you should be able to:

    1. *Describe the difference between content and presentation in web pages.*
    2. *Describe how styles are used to apply presentation to HTML tags.*
    3. *Describe how specificity of CSS rules affect how a page is rendered by a web browser.*
    4. *Use the CSS box model to determine size of HTML elements on the page.*
    5. *Use common CSS formatting rules for web pages.*
    6. *Apply CSS rules to affect the flow of content on a page - positioning.*
    7. *Use CSS for layout of an entire web page using Flex Box and Grid Layout.*

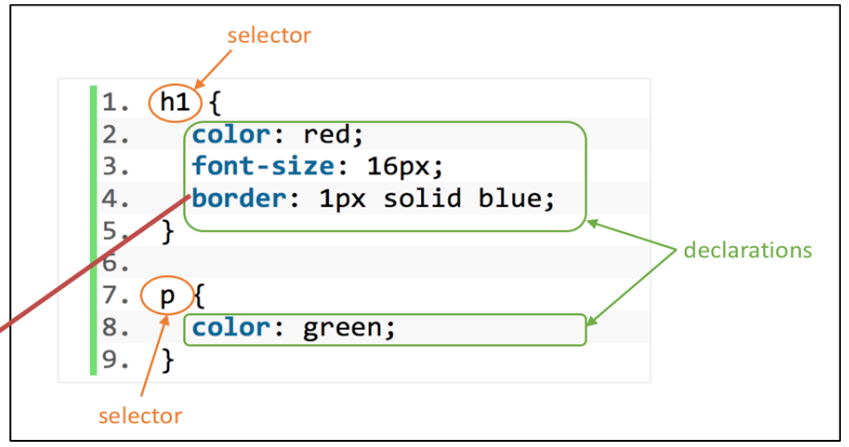
Separation of Content/Structure from Presentation

- *Web standards* — the importance of separating *structure* from *presentation*
- Controlling the presentation of a document with CSS becomes much easier if structure and presentation are separated

- **Structure** — the mandatory parts of an HTML document plus the semantic and structured markup of its contents.
- **Presentation** — the style you give the content — it is usually the way a document looks, but it can also affect *how a document sounds* — not everybody uses a graphical web browser. CSS is the standard focusing on styling HTML documents.

End: Section 1 - Introduction

Section 2: CSS Style and Rules




```
1. border-width: 1px;
2. border-style: solid;
3. border-color: blue;
```

- CSS Declarations:
  - *Properties and Values*
  - *Shorthand*

Common CSS Properties — I

Property	Description
background	Shorthand for setting a number of properties related to the *background* of an element (e.g. *-color, -image, -size, -repeat*). Often used to place images in the background of an element.
border	Used to set the border for an element. Allows shorthand for *-width, -style,* and *-color<'em>.*
border-radius	Allows for specifying a rounded border around an element.
bottom	Specifies an element's vertical position. The effect depends on how the element is positioned.
color	Sets the color of an element. Can be applied using named colors, hex values, RGB, or HSL values.
column-count	Divides an element into multiple columns.
display	Specifies the way content is rendered in an element. Allows for specifying many different types of rendering, such as inline (default), block, flow, and contents. By specifying none, the element is not displayed at all.
filter	Allows for applying graphical filters, such as blur, contrast, drop-shadow, invert, and saturation to an image.
font	Used to specify the font used for a text element. Can specify a list of increasingly available fonts, for example:

Property	Description
	font-family: Helvetica, Arial, sans-serif You can also specify font-style, which can set italics, font-weight to set bold, and font-size, which can set the size in pixels, em, or percent.

Common CSS Properties — II

Property	Description
height, width	Sets the height or width of an element in percent or pixels.
hyphens	Allows you to control how hyphens are added to text at line breaks. You can choose manual, none, or auto. In the auto mode, the browser uses the language you specify for the text to use its built-in rules to hyphenate words.
left, right, top, bottom	These two properties help specify the horizontal and vertical positioning of an element. The functioning depends on which positioning is used (e.g. absolute, relative etc). for example, if an element's position is set to absolute or fixed, the left/right/top/bottom properties specify the distance between the edge of the element and the corresponding edge of its containing block.
list-style	This allows you to specify how a list is rendered. You can control the shape of bullets (circles, squares, images) or the type of numbering used
margin	Using this property, you can control the space on all four sides of an element.
opacity	With this property, you can set how transparent an element is. This is often used with images.
overflow	You can decide what to do with content that doesn't fit in a containing element. Options include adding scrollbars, clipping, or displaying content overflowing from the container.
padding	Sets the padding area around an element. It uses shorthand to set each of the four sides of an element (-top, -right, -bottom, and -left).

Property	Description
position	Specifies the type of positioning used to place an element (static, relative, absolute, fixed, or sticky).

Common CSS Properties — III

Property	Description
text-align	Specifies how text is aligned (justified)
top, bottom	These properties work in concert with a specific positioning property to specify where the top or bottom of an element is rendered. The specific effect depends on which positioning is used. For example, if position is absolute or fixed, top sets the top edge of the element above/below the top edge of its nearest positioned ancestor, whereas if position is relative, top specifies how far the top edge is moved from its normal position.
vertical-align	Vertical alignment of an element (baseline, top, middle, bottom, …)
visibility	Specifies whether an element is hidden or visible.
z-index	Sets the stack order of overlapping elements. The higher the z-index value, the further to the top an element is placed.

End: Section 2 - CSS Styles and Rules

Section 3: CSS Selectors

- **CSS selector** selects the HTML element(s) you want to style - they are used to "find" (or select) the HTML elements you want to style.
- Five categories of CSS selectors:
  - *Simple selectors (select elements based on name, id, class)*
  - *Combinator selectors (select elements based on a specific relationship between them)*
  - *Pseudo-class selectors (select elements based on a certain state)*
  - *Pseudo-elements selectors (select and style a part of an element)*
  - *Attribute selectors (select elements based on an attribute or attribute value)*

Simple Selectors

Selector	Example	Example description
#id	#firstname	Selects the element with id="firstname"
.class	.intro	Selects all elements with class="intro"
element.class	p.intro	Selects only <p> elements with class="intro"
*	*	Selects all elements
element	p	Selects all <p> elements
element,element,..	div, p	Selects all elements **and** all <p> elements
ancestor descendant	section p …	The " " (space) combinator selects all the <p> elements that are descendants of the <section> element.

Class Selectors - Example

- CSS rules with two classes, namely, .ingredient and .measurement

```
.ingredient {
  color: blue;
}
.measurement {
  font-weight: bold;
  background-color: #dff5d6;
}
```

The following HTML code uses the above CSS style rules:

```
<ul>
  <li class="ingredient"><span class="measurement">&frac12; tsp</span> cinnamon</li>
  <li class="ingredient"><span class="measurement">1 cup</span> sugar</li>
</ul>
```

The following is the browser rendering of the CSS style rules in the above code:

- **½ tsp** cinnamon
- **1 cup** sugar

Multiple Selectors for a CSS Rule

```
h1, h2, h3, h4{
  color: green;
  font-size:12;
  font-family: Verdana, Arial, Sans-serif;
}

p, .ingredient{
  color: blue;
}
```

Linking CSS stylesheet to HTML

An external CSS style sheet is linked to HTML document inside the <head> element as follows:

```
<head>
  <meta charset="uft-8" />
  <title>Oatmeal Recipe</title>
  <link rel="stylesheet" type="text/css" href="css/styles.css" />
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/css/bootstrap.min.css" />
</head>
```

CSS Styled Recipe Webpage

P&I Recipes

Oatmeal | Pancakes | Dinner | Snacks

Related Recipes

Nutty & Fruity Oatmeal

A healthy breakfast is key to a helthy diet. There are many web sites that describe the benefits of oatmeal, including: WebMD Organic facts HHS

Why oatmeal?

Among the many benefits of oatmeal are:

- excellent source of fiber that helps to reduce cholestrol
- loaded with nutrients including vitamins, minerals, and antioxidants
- helps with weight reduction
- reduce constipation

Here is a recipe for a *delicious* oatmeal that increases the high nutritious value of plain oatmeal.

Ingredients

- 1/2 cup quick oats
- 1/2 cup each of almond and vanila soy milk:
- 2 Tbsp chopped walnuts
- 1/2 cup each chopped banana and blueberries
- A pinch each of cardamom and ginger powder
- 1/4 tsp cinnamon powder

Not sure what quick oats are? Quick oats are oat grains that are cut to smaller pieces and then steamed and rolled.

Directions

1. Mix all ingredients in a cooking pot
2. Add one cup of water
3. Bring to a boil and simmer for a couple of minutes. **Important:** Make sure you stir the mixture **frequently** while simmering.

Nutritional value

Nutritional value of 1/2 cup of dry quick oats

Nutrition	Amount
Calories	150 cal (kcal)
Carbohydrates	27 g
Protein	5 g
Fat	3 g
Dietary fiber	4 g

You may find the nutritional value of almost any food from sources like Nutrition Value and What's In Food

Contacts: J. Iversen, iversen@uwosh.edu; G. Philip, philip@uwosh.edu

Visit the styled P & I Recipe Website

End: Section 3 - CSS Selectors

Section 4: Cascading Style: Concept & Algorithm — I

- The **cascade** is an *algorithm* that defines how user agents (browsers) combine CSS property values originating from different sources.
  - *The cascade defines the **origin** and **layer** that takes precedence when declarations in more than one origin or cascade layer set a value for a property on an element.*
- CSS declarations come from different origin types, which are:

  1. User-agent stylesheets - set in the browser

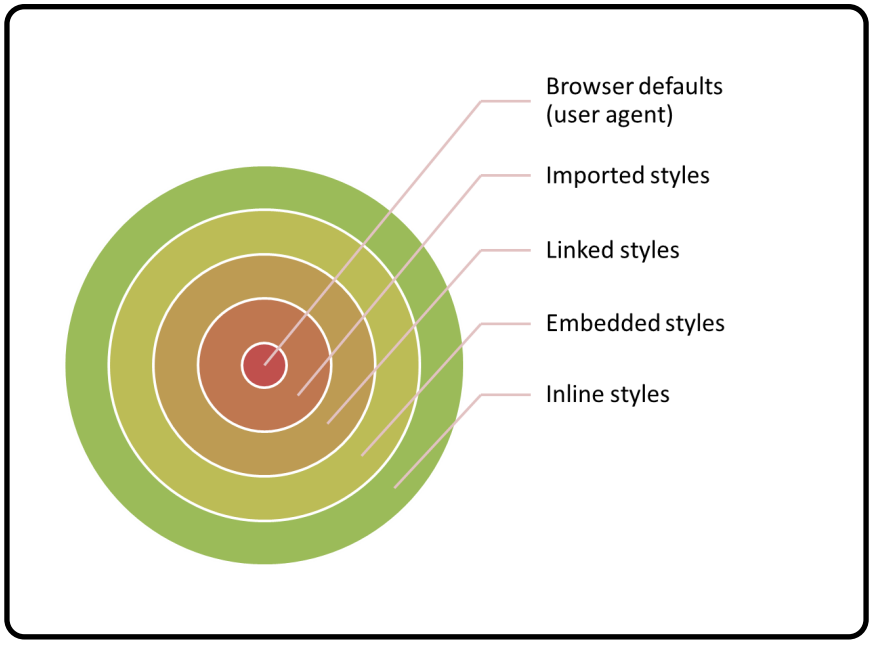
  2. Author stylesheets - created by Web developer

  3. User stylesheets - set by reader of Web browser

Cascading Style: Concepts & Algorithm — II

- Opposite diagram illustrates the **cascade layers** with precedence decreasing as one goes from outer to inner layers:

    1. ***Inline styles:*** *Written directly into the page (and tags). Inline styles control a single character, a line, or short areas of that page.*
    2. ***Embedded styles:*** *Appears in the head portion of the page, and provides named styles for that particular HTML document*
    3. ***Linked styles:*** *External style sheets. A link appears in the head portion of the HTML pages that are to be controlled by a particular linked style sheet.*
    4. ***Imported styles:*** *A style sheet referenced from within another style sheet.*



For greater detail, read the following article: *Mozilla Developer Network (MDN), Introducing the CSS Cascade, June 12, 2022*

5. ***Browser defaults*** *(user agent): When no style is available, the browser defaults are used.*

Cascading Algorithm Steps

- This algorithm runs in the browser (user agent).

1. **Elevance:** Filter all the rules from the different sources to keep only the rules that apply to a given element.
2. **Origin and Importance:** Sort these rules according to their importance, that is, whether or not they are followed by `!important`, and by their origin.
3. **Specificity:** In case of *conflict* (i.e., equality with an origin), the specificity of a rule is considered to choose one value or another.
4. **Order of appearance:** If there is conflict within an origin (i.e., there are competing values for a property that are in style block matching selectors of equal specificity), then the last declaration in the style order is applied

Specificity Rules

- Points assigned to CSS rules:
  - *Inline style attributes: 1,000 points*
  - *ID rules: 100 points*
  - *Attribute, class or pseudo-class: 10 points*
  - *Element name or pseudo-element: 1 point*
- Highest point value rule that matches a given element "wins"

Example:

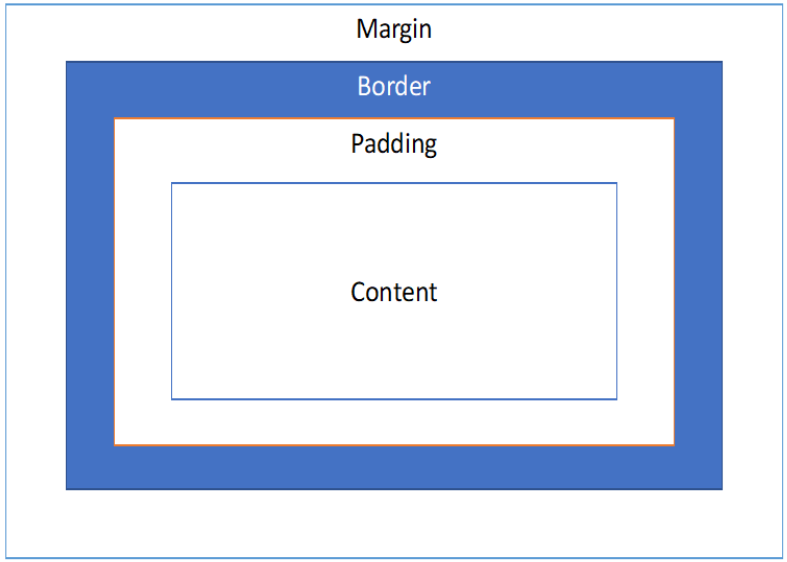
```
section img:hover .profile #directions {
  background-color: blue
}
```

- Analysis of the points scheme of this rule:
  - *section: 1 pt (element)*
  - *img: 1 pt (element)*
  - *hover: 10 pts (pseudo class)*
  - *profile: 10 pts (class)*
  - *directions: 100 pts (ID*
  - *Total:* ***122 points***

End: Section 4 - Cascading Style Algorithm

Section 5: CSS Box Model

CSS Property	Size Attributes	Notes
Content	width, height, top, bottom.	Only sets the size for the content, not the entire box. Can also set min and max values for width and height.
Padding	padding-top, padding-bottom, padding-left, padding-right	
Border	border-width	Can also set border-style, border-color, and specify rounded



		corners.
Margin	margin-top, margin-bottom, margin-left, margin-right	

Specifying the CSS box values

Number of values Specified	Example	Result Decription
1	border-width: 10px;	Same size on all four sides of the box
2	border-width: 10px 20px;	Different vertical and horizontal values – first value specifies the vertical (top and bottom)
3	border-width: 10px 20px 30px;	Different top and bottom and same values for left and right in the order of top, horizontal, bottom
4	border-width: 10px 20px 30px 40px;	Different values on all four sides, in the order of top, right, bottom, left (TRBL – trouble)

Meassurement Units

- **Length:** Use any number of measures, such as pixels (px), cm, mm, em, rem, inches (in) etc. The default is pixels.
- **Percent** *(not valid for the border):* Sets the size as a percentage of the containing element.
- **Auto** *(not valid for the border):* The browser calculates the size automatically.
- **Inherit** *(margin and padding only):* The length is inherited from the parent element.
- **Tip:** The em and rem units are practical in creating perfectly scalable layout!

Absolute and Relative Measurements

Absolute Measurements

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

- Absolute length units are fixed.
- Length expressed in any of these will appear as exactly that size.
- Absolute length units are not recommended for use on screen, because screen sizes vary so much.
  - *But, they can be used if the output medium is known, such as A4 size for print layout.*

Relative Measurements

Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element

- Relative length units specify a length relative to another length property.
- Relative length units scale better between different rendering medium.

End: Section 5 - CSS Box Model

Section 6: CSS Positioning

Normal Flow of HTML

- Each HTML element is rendered one after the other horizontally.
- Browser decides line breaks – unless the element forces a line break (e.g. <p>)
- Use as much as possible

Floating

- The `float` property specifies whether an element should float to the **left**, **right**, or not at all.
- Note the following:

  1. *Absolutely positioned elements ignore the float property!*

Examples

- Example for making an image float to the right:

```
img  {
  float: right;
}
```

- Let image be displayed just where it occurs in the text (float: none):

```
img  {
  float: none;
}
```

- Do not allow floating elements on the left or the right side of a specified <p> element:

```
img {
  float: left;
}

p.clear {
  clear: both;
}
```

- If a floating element is taller than the containing element, it will overflow outside its container. *It is possible to fix this with the "clearfix hack" as follows:*

```
.clearfix::after {
  content: "";
```

2. *Elements next to a floating element will flow around it.*

- *To avoid this, use the `clear` property or the `clearfix` hack (see examples).*

```
  clear: both;
  display: table;
}
```

The ‘position’ Property

Values of ‘position’ Property	Effect of Value
static	Default value. Elements are placed based on normal flow.
relative	The element is placed relative to its normal position – you can specify how far away in pixels, em etc. the element should be placed from where it would normally be.
fixed	Places the item in a fixed position on the screen, regardless of how the page is scrolled. You can control where on the screen an item is placed – for instance, you can have an item be fixed at the bottom of the screen.
absolute	Positioned relative to the nearest positioned ancestor (an element placed outside the element). If an absolute element doesn’t have any positioned ancestor it will use the document body.
sticky	This is the newest addition to the positioning family and isn’t always well supported. But it allows for an element that can switch between relative and fixed depending on scroll position. It is often used to have an element scroll up the screen until it reaches a specific point, and then it “sticks”. For example, a menu bar may start below a header image, but then as the user scrolls down, the menu bar sticks to the very top of the screen.

Pseudo Classes

- "pseudo-class" — not really full classes

- Style links based on **state**

```
a:link {/* an unvisited link*/
  color: blue;
}
a:visited {/** a visited link/
  colour: green;
}
a:hover {/* when hovering over a link happens */
  color: red;
}
a:active {/* an activated link */
  color: purple;
}
```

- Selecting elements based on position relative to other elements

```
li:first-child {/* the first element in a list */
  color: red;
}
li:last-child {/* the last element in a list */
  color: red;
}
li:nth-child(odd) {/* all odd elements in a list */
  color: red;
}
li:nth-child(4) {/* element number 4 in a list */
  color: red;
}
li:nth-child(3n) {/* elements 3, 6, 6, etc in a list
(n = 0, 1, 2, ...) */
  color: red;
}
li:nth-child(3n+2) {/* (3n+2)th element in a list,
where n = 0, 1, 2, ... */
  color: red;
}
```

Pseudo-elements

- A CSS pseudo-element is indicated by a keyword added to a selector that lets you style a specific part of the selected element(s).
- In contrast to pseudo-elements, pseudo-classes can be used to style an element based on its state.
- Syntax for pseudo-elements:

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

- Example: `::first-line` can be used to change the font of the first line of a paragraph.

```
/* The first line of every <p> element. */
p::first-line {
  color: blue;
  text-transform: uppercase;
```

- It's possible to combne pseudo-classes with pseudo-elements:

```
article p:first-child::first-line {
  font-size: 120%;
  font-weight: bold;
}
```

- The `::before` and `::after` pseudo-elements enable you to insert content into the document using CSS

```
.box::after {
    content: " ➥"
}
```

- Use in HTML:

```
<p class="box">Content in the box in my HTML page.</p>
```

- Outcome in a browser:

Content in the box in my HTML page. ➥

```
}
```

CSS Functions

- Functions are usually associated with languages like JavaScript, Python, or C++, but they do exist in CSS too, as **property values**.
- We've already seen functions in action, e.g., the `nth-child()` function (although this is atypical in that it's not a property value. Instead, its the element whose property is to be styled).
- Example:
  - *We illustrate using `calc()` to make the box 20% + 100px wide.*
  - *The 20% is calculated from the width of the parent container `.wrapper` and so will change if that width changes.*
  - *We can't do this calculation beforehand because we don't know what 20% of the*

- CSS code *(example cont'd):*

```css
.wrapper {
  width: 400px;
}
.mybox {
  padding: 1em;
  border-radius: .5em;
  border: 5px solid rebeccapurple;
  background-color: lightblue;
  width: calc(20% + 100px);
}
```

- HTML code:

```html
<div class="wrapper">
  <div class="mybox">My box width is calculated.</div>
</div>
```

- Result in a browser:

My box width is calculated.

parent will be, so we use the function `calc()` *to tell the browser to do it for us.*

- *This exampe is presented to the* ***right of this slide****.*

- Further reading:
  - *Eric Bailey, A Complete Guide to CSS Functions, May 4, 2020 (Updated on Oct 15, 2021), Accessed: 2022-07-15*
  - *QuackIt.com, CSS Functions, Accessed: 2022-07-15*

End: Section 6

Section 7: Formatting Tables Using CSS

- HTML code for the Nutrition Table

```
<table id="nutrition-table" border="1" style="color: chocolate; font-size: 14px">
  <caption>Nutritional value of &frac12; cup of dry quick oats</caption>
  <tr>
    <th>Nutrition</th>
    <th>Amount</th>
  </tr>
  <tr>
    <td>Calories</td>
    <td>150 cal (kcal)</td>
  </tr>
  <tr>
    <td>Carbohydrates</td>
    <td>27 g</td>
  </tr>
  <tr>
    <td>Protein</td>
    <td>5 g</td>
  </tr>
  <tr>
    <td>Fat</td>
    <td>3 g</td>
  </tr>
  <tr>
    <td>Dietary fiber</td>
    <td>4 g</td>
  </tr>
</table>
```

- CSS code for styling the Nutrition Table

```
#nutrition-table {
  border: 3px solid black; /* Outside border */
  width: 300px;
  border-collapse: collapse;
}
#nutrition-table th, td {
  border: 1px solid black; /* Inside borders */
  padding: 0px 5px;
}
#nutrition-table tr:nth-child(even) { /* Zebra stripes */
  background-color: lightgray;
}
#nutrition-table td:nth-child(1) { /* First column */
  text-align: right;
}
#nutrition-table caption {
  font-style: italic;
  font-size: small;
}
```

- The Styled Nutrition table:

Nutritional value of 1/2 cup of dry quick oats

Nutrition	Amount
Calories	150 cal (kcal)
Carbohydrates	27 g
Protein	5 g
Fat	3 g
Dietary fiber	4 g

- Also take a look at it on the P&I Recipe Website.

Page Layout Using CSS Flexbox

HTML:

```html
<div class="flexbox">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
</div>
```

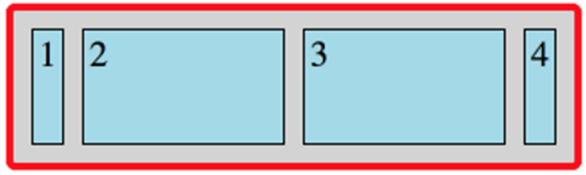
Output:

CSS:

```css
.flexbox {
    display: flex;
    border: 4px solid red;
    border-radius: 5px;
    width: 300px;
    height: 75px;
    background-color: lightgray;
    padding: 5px;
}

.flexbox > div{
    border: 1px solid black;
    background-color: lightblue;
    font-size: 20;
    padding: 3px;
    margin: 5px;
}
```

```css
.flexbox :nth-child(2){
    flex: 1 1 200px;
}
```

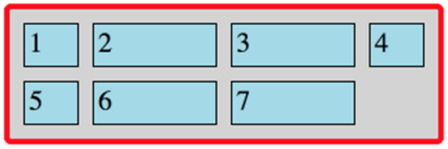


Flex Item Properties

Flex Item Property	Description
flex-grow	If set to a positive value, the item is allowed to grow to take up available space in the flexbox.
flex-shrink	If set to a positive value, the item is allowed to shrink if there isn't enough space inside the flexbox to contain all the elements at their basis size.
flex-basis	The size of the item along the main axis of the flexbox. If all the elements fit exactly the flexbox with this size, flex-grow and flex-shrink are not applied.

Grid Layout

```html
<div class="gridcontainer">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
    <div>5</div>
    <div>6</div>
    <div>7</div>
</div>
```




```css
.gridcontainer {
    display: grid;
    grid-template-columns: 1fr 2fr 2fr 1fr;
    border: 4px solid red;
    border-radius: 5px;
    width: 300px;
    background-color: lightgray;
    padding: 5px;
}

.gridcontainer > div {
    border: 1px solid black;
    background-color: lightblue;
    font-size: 20;
    padding: 3px;
    margin: 5px;
}
```

End: Section 7l

Section 8: Topic Summary

- In this topic, we have covered CSS especially the following aspects of CSS:
  - *Introduction to Styling Using CSS*
  - *Style Conflict Resolution: precedence and specificity*
  - *CSS Box Model*
  - *CSS Positioning*
  - *Pseudo Classes and Elements*
  - *Page Layout with HTML and CSS, including table styling*
