CSCI-4208: Developing Advanced Web Applications

Week 1: Lecture 2

Overview

Unit 1: Frontend development → CSS Crash Course

1. What is CSS
2. Importing CSS into a HTML document
3. CSS Syntax
4. CSS Selectors
5. CSS Colors
6. CSS Fonts
7. CSS Spacing
8. CSS Backgrounds
9. HTML Element Customizations
10. CSS Aligning: Flexbox
11. CSS Aligning: Grid

1. What is CSS?

- CSS stands for Cascading Style Sheets
- CSS is a declarative styling language
- CSS describes how HTML elements are to be displayed.
- CSS saves a lot of work. It can control the layout of multiple web pages all at once. It allows the DRY principle to apply toward styling.
- External stylesheets are stored in CSS files

1. CSS Solved a Big Problem for HTML

- HTML was NEVER intended to contain tags for formatting a web page!
- HTML was created to describe the content of a web page, like:
  - `<h1>This is a heading</h1>`
  - `<p>This is a paragraph.</p>`
- CSS is responsible for the style formatting for an HTML page!

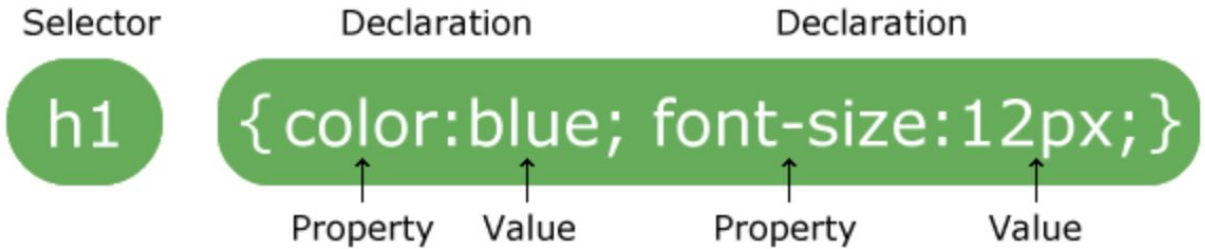
2. Importing CSS into a HTML document

Link your CSS files in the <head> element of the HTML document.

```
<html>
    <head>
        <link href='styles/styles.css' rel="stylesheet">
    </head>
     <body>...</body>
</html>
```

3. CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

4. CSS Selectors

CSS selectors are used to "find" (or select) the HTML elements you want to style.

There are three basic types of CSS Selectors:

- **Element selectors** *(select elements based on element type)*
- **Class selectors** *(select elements based on class attribute)* ← ***Industry standard***
- **ID selectors** *(select elements based on id attribute)*

More advanced types of CSS Selectors:

- **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)*

These advanced types won't be covered in detail for this course, check MDN or W3Schools for more info.
Note: They do appear in Lab 2 within the 'HTML Element Customization' section.

4. CSS Selectors - Examples

index.html

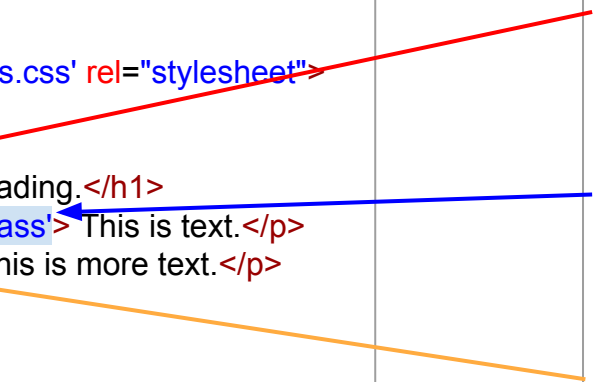
```
<html>
   <head>
       <link href='styles.css' rel="stylesheet">
   </head>
   <body>
      <h1>This is a heading.</h1>
      <p class='html-class'>This is text.</p>
      <p id='html-id'>This is more text.</p>
   </body>
</html>
```

styles.css

```
/*Element Selector*/
h1 {
        color:  red;
}

/*Class Selector*/
.html-class {
        color: blue;
}

/*ID Selector*/
#html-id {
        color: yellow;
}
```



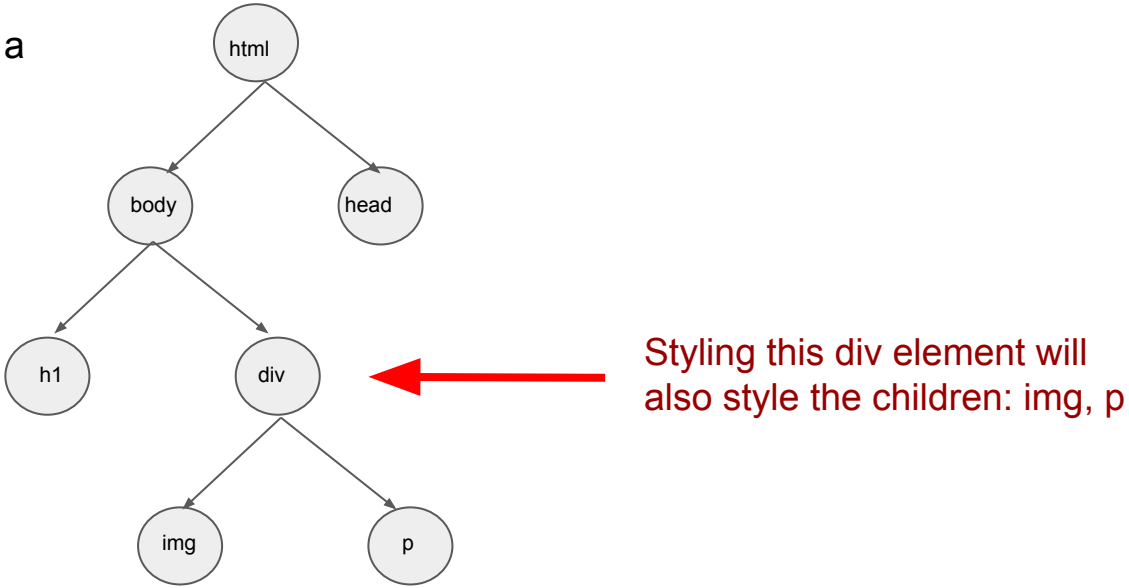
4. All Simple CSS Selectors

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

4. Cascading Order

HTML documents are modeled as a trees of elements.

A CSS style cascades from the parent element to all of its children



4. Cascading Order

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

5. CSS Colors - Overview

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

- Predefined Color Name: All browsers support 140 color names
- RGB: a decimal encoding for each primary color: red, green, blue (from light color theory)
- HEX: a hexadecimal encoding for the RGB decimal representation
- HSL: a decimal encoding for attributes: Hue, Saturation, Lightness
- RGBA: RGB encoding with an alpha transparency value between 0-1
- HSLA: HSL encoding with an alpha transparency value between 0-1

5. CSS Colors: RGB Color Encoding

RGB is a decimal encoding for each primary color: red, green, blue (in light color theory)

Red: 0-255, Green: 0-255, Blue: 0-255, **Formatted: RGB(0,0,0) - RGB(255,255,255)**

The combination of Red, Green & Blue values gives more than 16 million different colors (**256 x 256 x 256**)

Examples:

Primary colors	Secondary colors	Grayscale
Red: RGB(255,0,0) Green: RGB(0,255,0) Blue: RGB(0,0,255)	Yellow: RGB(255,255,0) Fuchsia: RGB(255,0,255) Aqua: RGB(0,255,255)	Black: RGB(0,0,0) White: RGB(255,255,255)

5. CSS Colors: HEX Color Encoding

HEX is a hexadecimal encoding for the RGB decimal representation of encoding colors

Red: 00-FF, Green: 00-FF, Blue: 00-FF, **Formatted: #000000 - #FFFFFF**

The combination of Red, Green & Blue values gives more than 16 million different colors (**256 x 256 x 256**)

Examples:

Primary colors	Secondary colors	Grayscale
Red: #FF0000	Yellow: #FFFF00	Black: #000000
Green: #00FF00	Fuchsia: #FF00FF	White: #FFFFFF
Blue: #0000FF	Aqua: #00FFFF	

5. CSS Colors: HSL Color Encoding

HSL stands for hue, saturation, and lightness.

- Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, 240 is blue.
- Saturation is a percentage value; 0% means a shade of gray and 100% is the full color.
- Lightness is also a percentage; 0% is black, 100% is white.

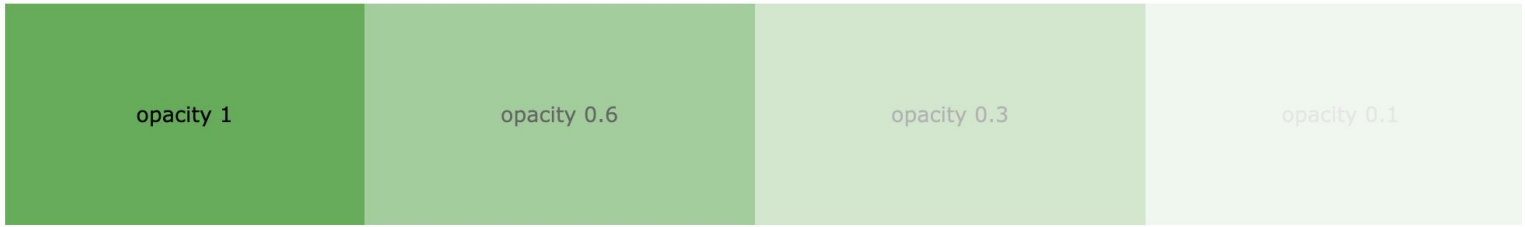
Examples:

Primary colors	Secondary colors	Grayscale
Red: hsl(0, 100%, 50%) Green: hsl(120, 100%, 50%) Blue: hsl(240, 100%, 50%)	Yellow: hsl(60, 100%, 50%) Fuchsia: hsl(300, 100%, 50%) Aqua: hsl(180, 100%, 50%)	Black: hsl(0, 0%, 0%) White: hsl(0, 0%, 100%)

5. CSS Colors: Alpha value

Opacity / Transparency

The `opacity` property specifies the opacity/transparency of an element. It can take a value from 0.0 - 1.0. The lower value, the more transparent:



5. CSS Colors: 140 Predefined Color Names

Green Colors

Color Name	HEX	Color
GreenYellow	#ADFF2F	
Chartreuse	#7FFF00	
LawnGreen	#7CFC00	
Lime	#00FF00	
LimeGreen	#32CD32	
PaleGreen	#98FB98	
LightGreen	#90EE90	
MediumSpringGreen	#00FA9A	
SpringGreen	#00FF7F	
MediumSeaGreen	#3CB371	
SeaGreen	#2E8B57	
ForestGreen	#228B22	
Green	#008000	
DarkGreen	#006400	
YellowGreen	#9ACD32	
OliveDrab	#6B8E23	
DarkOliveGreen	#556B2F	
MediumAquaMarine	#66CDAA	
DarkSeaGreen	#8FBC8F	
LightSeaGreen	#20B2AA	
DarkCyan	#008B8B	
Teal	#008080	

Cyan Colors

Color Name	HEX	Color
Aqua	#00FFFF	
Cyan	#00FFFF	
LightCyan	#E0FFFF	
PaleTurquoise	#AFEEEE	
Aquamarine	#7FFFD4	
Turquoise	#40E0D0	
MediumTurquoise	#48D1CC	
DarkTurquoise	#00CED1	

Red Colors

Color Name	HEX	Color
LightSalmon	#FFA07A	
Salmon	#FA8072	
DarkSalmon	#E9967A	
LightCoral	#F08080	
IndianRed	#CD5C5C	
Crimson	#DC143C	
Red	#FF0000	
FireBrick	#B22222	
DarkRed	#8B0000	

5. CSS Colors: 140 Predefined Color Names

Purple Colors

Color Name	HEX	Color
Lavender	#E6E6FA	
Thistle	#D8BFD8	
Plum	#DDA0DD	
Orchid	#DA70D6	
Violet	#EE82EE	
Fuchsia	#FF00FF	
Magenta	#FF00FF	
MediumOrchid	#BA55D3	
DarkOrchid	#9932CC	
DarkViolet	#9400D3	
BlueViolet	#8A2BE2	
DarkMagenta	#8B008B	
Purple	#800080	
MediumPurple	#9370DB	
MediumSlateBlue	#7B68EE	
SlateBlue	#6A5ACD	
DarkSlateBlue	#483D8B	
RebeccaPurple	#663399	
Indigo	#4B0082	

Pink Colors

Color Name	HEX	Color
Pink	#FFC0CB	
LightPink	#FFB6C1	
HotPink	#FF69B4	
DeepPink	#FF1493	
PaleVioletRed	#DB7093	
MediumVioletRed	#C71585	

Yellow Colors

Color Name	HEX	Color
Gold	#FFD700	
Yellow	#FFFF00	
LightYellow	#FFFFE0	
LemonChiffon	#FFFACD	
LightGoldenRodYellow	#FAFAD2	
PapayaWhip	#FFEFD5	
Moccasin	#FFE4B5	
PeachPuff	#FFDAB9	
PaleGoldenRod	#EEE8AA	
Khaki	#F0E68C	
DarkKhaki	#BDB76B	

5. CSS Colors: 140 Predefined Color Names

Blue Colors

Color Name	HEX	Color
CadetBlue	#5F9EA0	
SteelBlue	#4682B4	
LightSteelBlue	#B0C4DE	
LightBlue	#ADD8E6	
PowderBlue	#B0E0E6	
LightSkyBlue	#87CEFA	
SkyBlue	#87CEEB	
CornflowerBlue	#6495ED	
DeepSkyBlue	#00BFFF	
DodgerBlue	#1E90FF	
RoyalBlue	#4169E1	
Blue	#0000FF	
MediumBlue	#0000CD	
DarkBlue	#00008B	
Navy	#000080	
MidnightBlue	#191970	

Brown Colors

Color Name	HEX	Color
Cornsilk	#FFF8DC	
BlanchedAlmond	#FFEBCD	
Bisque	#FFE4C4	
NavajoWhite	#FFDEAD	
Wheat	#F5DEB3	
BurlyWood	#DEB887	
Tan	#D2B48C	
RosyBrown	#BC8F8F	
SandyBrown	#F4A460	
GoldenRod	#DAA520	
DarkGoldenRod	#B8860B	
Peru	#CD853F	
Chocolate	#D2691E	
Olive	#808000	
SaddleBrown	#8B4513	
Sienna	#A0522D	
Brown	#A52A2A	
Maroon	#800000	

5. CSS Colors: 140 Predefined Color Names

White Colors

Color Name	HEX	Color
White	#FFFFFF	
Snow	#FFFAFA	
HoneyDew	#F0FFF0	
MintCream	#F5FFFA	
Azure	#F0FFFF	
AliceBlue	#F0F8FF	
GhostWhite	#F8F8FF	
WhiteSmoke	#F5F5F5	
SeaShell	#FFF5EE	
Beige	#F5F5DC	
OldLace	#FDF5E6	
FloralWhite	#FFFAF0	
Ivory	#FFFFF0	
AntiqueWhite	#FAEBD7	
Linen	#FAF0E6	
LavenderBlush	#FFF0F5	
MistyRose	#FFE4E1	

Grey Colors

Color Name	HEX	Color
Gainsboro	#DCDCDC	
LightGray	#D3D3D3	
Silver	#C0C0C0	
DarkGray	#A9A9A9	
DimGray	#696969	
Gray	#808080	
LightSlateGray	#778899	
SlateGray	#708090	
DarkSlateGray	#2F4F4F	
Black	#000000	

Orange Colors

Color Name	HEX	Color
Orange	#FFA500	
DarkOrange	#FF8C00	
Coral	#FF7F50	
Tomato	#FF6347	
OrangeRed	#FF4500	

6. CSS Fonts - Overview

- Generic Font Families
  - Font examples
- Font properties
- Google Fonts
  - Google Fonts examples
  - Google Fonts Effects
  - Google Fonts Effects examples
- Font Pairing Guidelines

6. CSS Fonts: Generic Font Families

In CSS there are five generic font families:

- **Serif fonts** - have a small stroke at the edges of each letter.
  - They create a sense of formality and elegance.
- **Sans-serif fonts** - have clean lines (no small strokes attached).
  - They create a modern and minimalistic look.
- `Monospace fonts - here all the letters have the same fixed width.`
  - `They create a mechanical look.`
- Cursive fonts - imitate human handwriting.
  - They create a personal, hand-crafted look.
- ***Fantasy fonts -*** *are decorative fonts.*
  - *They create a whimsical or playful look.*

All the different font names belong to one of the generic font families.

6. CSS Fonts: Font Examples

Generic Font Family	Examples of Font Names
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Lucida Console Monaco
Cursive	Brush Script MT Lucida Handwriting
Fantasy	COPPERPLATE Papyrus

6. CSS Fonts: font properties

Property	Description	Expected Values
font	Sets all the font properties in one declaration	
font-family	Specifies the font family for text	Font name
font-size	Specifies the font size of text	pixels (px), em, percent (%), viewport width (vw)
font-style	Specifies the font style for text	normal, italic
font-variant	Specifies whether or not a text should be displayed in a small-caps font	normal, small-caps
font-weight	Specifies the weight of a font	normal, lighter, bold

6. CSS Fonts: Google fonts

What is Google fonts?

Google Fonts is a library of over a 1000 free licensed font families, an interactive web directory for browsing the library, and APIs for conveniently using the fonts via CSS and Android.

Where to find Google fonts?

https://fonts.google.com/

How to use Google fonts?

Just add a special style sheet link in the <head> section and then refer to the font in the CSS

6. CSS Fonts: Google Fonts example - Single font

Example

Here, we want to use a font named "Audiowide" from Google Fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide">
<style>
body {
  font-family: "Audiowide", sans-serif;
}
</style>
</head>
```

Result:

Audiowide Font

Lorem ipsum dolor sit amet.

123456790

6. CSS Fonts: Google Fonts example - Multiple fonts

Example

Request multiple fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide|Sofia|Trirong">
<style>
h1.a {font-family: "Audiowide", sans-serif;}
h1.b {font-family: "Sofia", sans-serif;}
h1.c {font-family: "Trirong", serif;}
</style>
</head>
```

Result:

Audiowide Font

Sofia Font

Trirong Font

6. CSS Fonts: Google Fonts - Effects

Effect	API Name	Class Name
Anaglyph	anaglyph	font-effect-anaglyph
Emboss	emboss	font-effect-emboss
Fire	fire	font-effect-fire
Fire Animation	fire-animation	font-effect-fire-animation
Neon	neon	font-effect-neon
Outline	outline	font-effect-outline
Shadow Multiple	shadow-multiple	font-effect-shadow-multiple
Three Dimensional	3d	font-effect-3d
Three Dimensional Float	3d-float	font-effect-3d-float

https://developers.google.com/fonts/docs/getting_started#enabling_font_effects_beta

6. CSS Fonts: Google Fonts - Effects - Single effect

Example

Add the fire effect to the "Sofia" font:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia&effect=fire">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
}
</style>
</head>
<body>

<h1 class="font-effect-fire">Sofia on Fire</h1>

</body>
```

Result:

6. CSS Fonts: Google Fonts - Effects - Multiple effects

Example

Add multiple effects to the "Sofia" font:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia&effect=neon|outline|emboss|shadow-multiple">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
}
</style>
</head>
<body>

<h1 class="font-effect-neon">Neon Effect</h1>
<h1 class="font-effect-outline">Outline Effect</h1>
<h1 class="font-effect-emboss">Emboss Effect</h1>
<h1 class="font-effect-shadow-multiple">Multiple Shadow Effect</h1>

</body>
```

Result:

6. CSS Fonts: Font Pairing Guidelines

1. Compliment

A great font combination should harmonize, without being too similar or too different.

2. Contrast is King

Two fonts that are too similar will often conflict. Example: Combining serif with sans serif is a well known combination. .

3. Choose Only One Boss

One font should be the boss. This establishes a hierarchy for the fonts on your page. This can be achieved by varying the size, weight and color.

Online resources:

Fontjoy - an online font pairing tool: https://fontjoy.com/

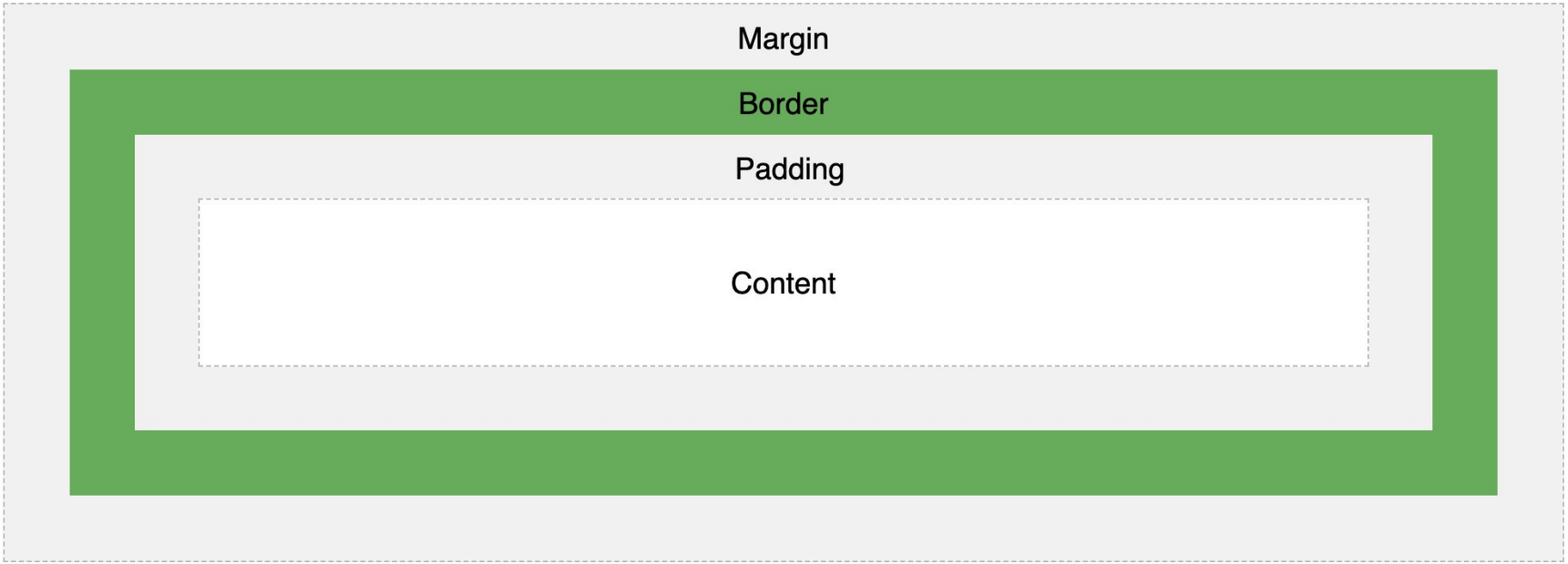
7. CSS Spacing - Overview

- Box Model
- Width & Height

7. CSS Spacing - Box Model

All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



7. CSS Spacing - Box Model

Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

7. CSS Spacing - Width & Height

CSS Setting height and width

The `height` and `width` properties are used to set the height and width of an element.

The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.

CSS height and width Values

The `height` and `width` properties may have the following values:

- `auto` - This is default. The browser calculates the height and width
- `length` - Defines the height/width in px, cm etc.
- `%` - Defines the height/width in percent of the containing block
- `initial` - Sets the height/width to its default value
- `inherit` - The height/width will be inherited from its parent value

Setting max-width

The `max-width` property is used to set the maximum width of an element.

The `max-width` can be specified in *length values*, like px, cm, etc., or in percent (%) of the containing block, or set to none (this is default. Means that there is no maximum width).

The problem with the `<div>` above occurs when the browser window is smaller than the width of the element (500px). The browser then adds a horizontal scrollbar to the page.

Using `max-width` instead, in this situation, will improve the browser's handling of small windows.

8. CSS Backgrounds

Property	Description
background	Sets all the background properties in one declaration
background-attachment	Sets whether a background image is fixed or scrolls with the rest of the page
background-clip	Specifies the painting area of the background
background-color	Sets the background color of an element
background-image	Sets the background image for an element
background-origin	Specifies where the background image(s) is/are positioned
background-position	Sets the starting position of a background image
background-repeat	Sets how a background image will be repeated
background-size	Specifies the size of the background image(s)

9. HTML Element Customizations

CSS provides powerful capabilities to change the default styles of HTML elements.

Examples include:

- Anchor elements can remove the underlining and blue color
- List can change the bullet point styles
- Tables can modify its row colors
- Buttons can change color with a hover action from mouse

For a complete list of customizations see:

https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

10. CSS Aligning: Flexbox

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structures.

Flexbox Elements

To start using the Flexbox model, you need to first define a flex container.

The element above represents a flex container (the blue area) with three flex items.

Resources:
https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout
https://www.w3schools.com/css/css3_flexbox.asp

10. CSS Aligning: Flexbox

Parent Element (Container)

Like we specified in the previous chapter, this is a flex **container** (the blue area) with three flex **items**:

The flex container becomes flexible by setting the `display` property to *flex*:

The flex container properties are:

- `flex-direction`
- `flex-wrap`
- `justify-content`
- `align-items`
- `align-content`

10. CSS Aligning: Flexbox - Properties

The flex-direction Property

The `flex-direction` property defines in which direction the container wants to stack the flex items.

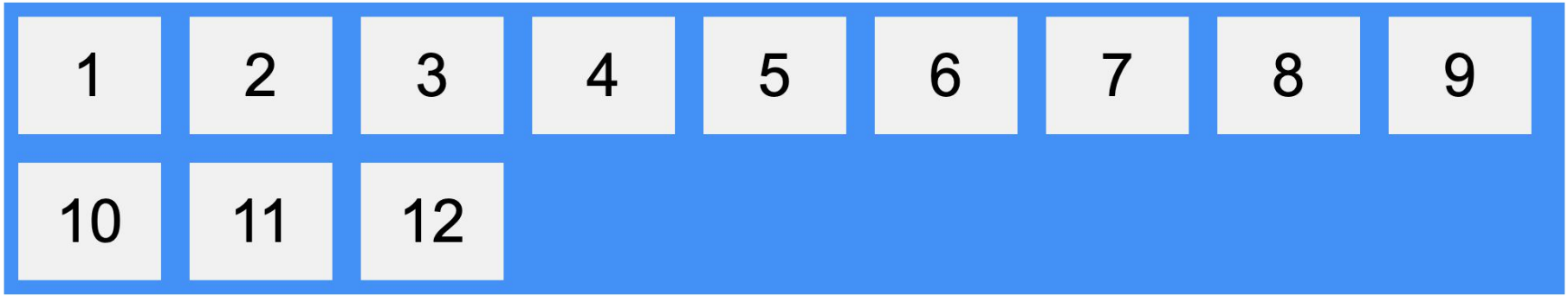
values: `column, column-reverse, row, row-reverse`

10. CSS Aligning: Flexbox - Properties

The flex-wrap Property

The `flex-wrap` property specifies whether the flex items should wrap or not.

The examples below have 12 flex items, to better demonstrate the `flex-wrap` property.



values: `wrap, nowrap, wrap-reverse`

10. CSS Aligning: Flexbox - Properties

The justify-content Property

The `justify-content` property is used to align the flex items:

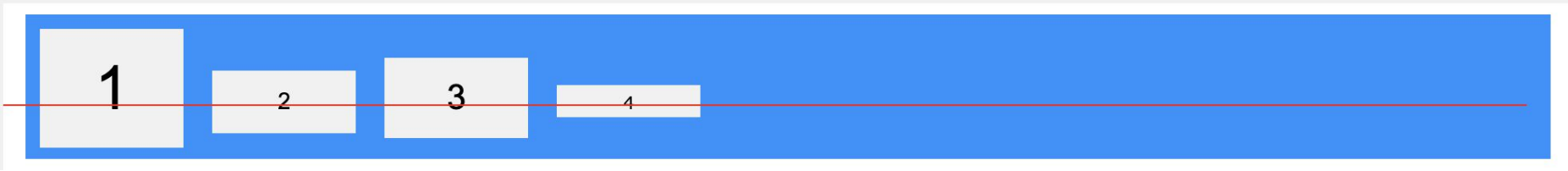
values: `center, flex-start, flex-end, space-around, space-between`

- The center value aligns the flex items at the center of the container
- The flex-start value aligns the flex items at the beginning of the container (this is default)
- The flex-end value aligns the flex items at the end of the container
- The space-around value displays the flex items with space before, between, and after the lines
- The space-between value displays the flex items with space between the lines

10. CSS Aligning: Flexbox - Properties

The align-items Property

The `align-items` property is used to align the flex items.



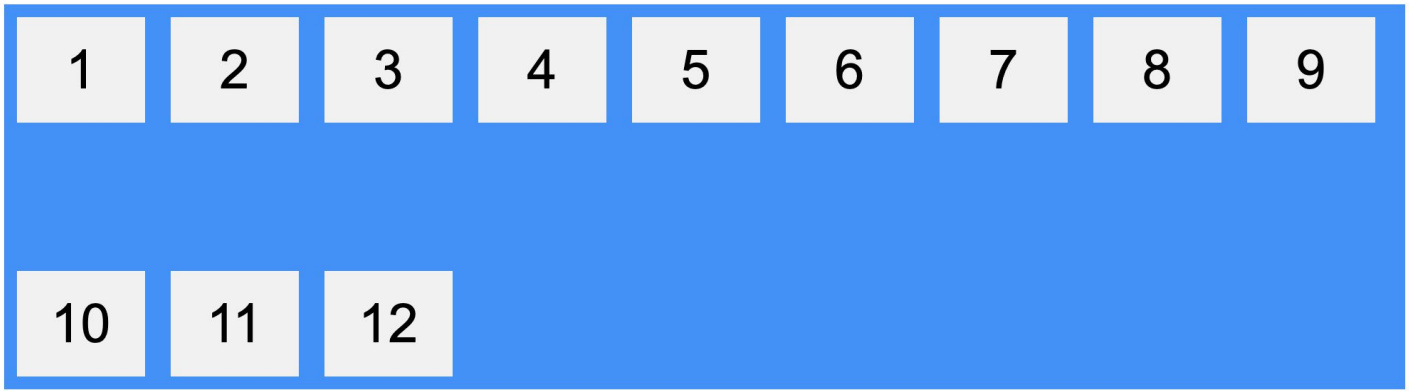
values: `center, flex-start, flex-end, stretch, baseline`

- The center value aligns the flex items in the middle of the container
- The flex-start value aligns the flex items at the top of the container:
- The flex-end value aligns the flex items at the bottom of the container:
- The stretch value stretches the flex items to fill the container (this is default):
- The baseline value aligns the flex items such as their baselines aligns:

10. CSS Aligning: Flexbox - Properties

The align-content Property

The `align-content` property is used to align the flex lines.



values: `space-between, space-around, stretch, center, flex-start, flex-end`

- The space-between value displays the flex lines with equal space between them:
- The space-around value displays the flex lines with space before, between, and after them:
- The stretch value stretches the flex lines to take up the remaining space (this is default):
- The center value displays display the flex lines in the middle of the container:
- The flex-start value displays the flex lines at the start of the container:
- The flex-end value displays the flex lines at the end of the container:

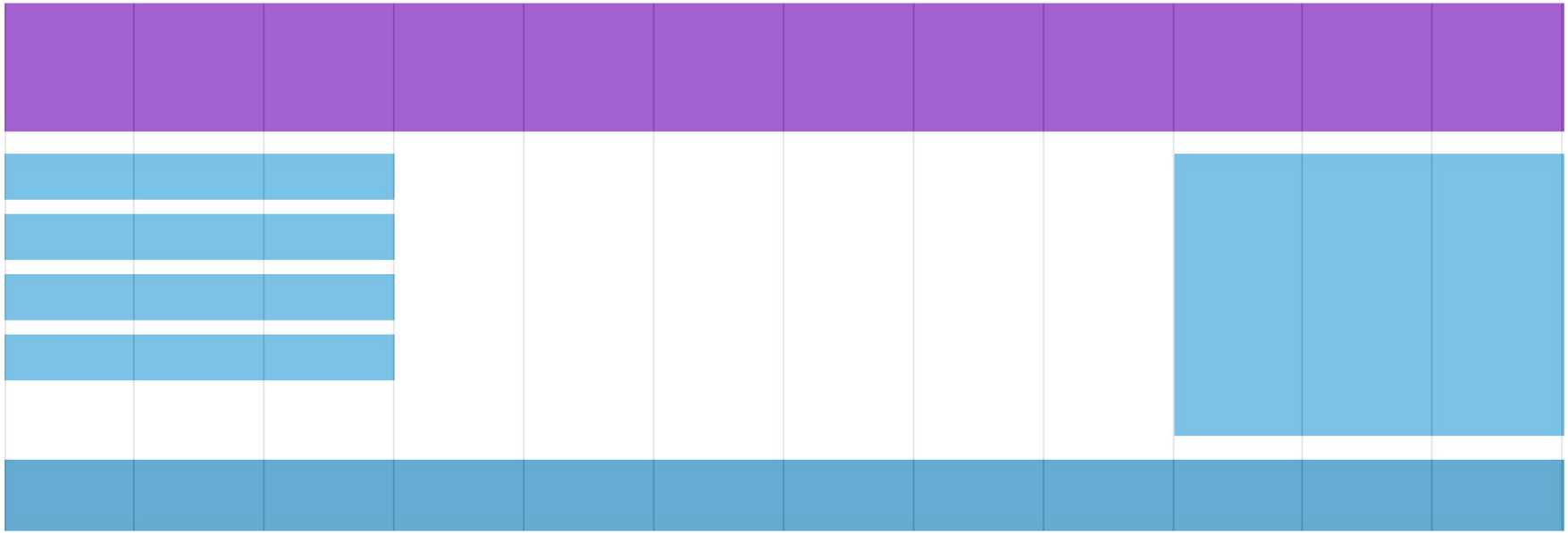
11. CSS Aligning: Grid

What is a Grid-View?

Many web pages are based on a grid-view, which means that the page is divided into columns:

11. CSS Aligning: Grid

Using a grid-view is very helpful when designing web pages. It makes it easier to place elements on the page.



The End.

Let's explain implementation details from *Lab 1: HTML*

Coming Soon... to a Moodle near you!

Lab 2	CSS Crash Course Lab
Homework 2	Style your *'Interactive Story'* with CSS