COMP 1537 Tutorial: Stylesheets and CSS

1. Introduction to CSS

When we build web pages, HTML provides the **content** and **structure**, while CSS (Cascading Style Sheets) is used to provide the **style**; the way things look.

This separation is important:

- **HTML** handles meaning and semantics.
- **CSS** handles visual presentation.

By keeping these separate, you can change the style of an entire website without touching the HTML content.

2. CSS Basics

CSS Files

- CSS is written in plain text files with the extension `.css`.
- Example file: `style.css`

```
li {
  font-style: italic;
  color: grey;
}
```

This will make every `<li>` (list item) display as **grey italic text**.

Linking CSS to HTML

To apply styles, include a `<link>` inside the `<head>` of your HTML:

```
<link rel="stylesheet" href="style.css" />
```

- `href` is usually a relative path but can also be an absolute URL.
- Reloading the page applies the new styles.

3. Anatomy of a CSS Rule

A CSS rule has three main parts:

```
li {
  font-style: italic;
  color: grey;
}
```

- **Selector**: `li`: what to style (all `<li>` elements).
- **Properties**: `font-style`, `color`: what aspect to change.
- **Values**: `italic`, `grey`: how the property should change.

You can have multiple rules in a single CSS file:

```css
h1 {
  text-decoration: underline;
  text-align: center;
}
```

3.b. Where to Put CSS

There are three ways to include CSS in your HTML:

1. **External stylesheet** (recommended):

   ```html
   <link rel="stylesheet" href="style.css" />
   ```

2. **Internal stylesheet** (within `<head>`):

   ```html
   <style>
     li {
       font-style: italic;
       color: grey;
     }
   </style>
   ```

3. **Inline styles** (directly on an element, not recommended):

   ```html
   <li style="font-style: italic; color: grey;">Item</li>
   ```

4. Common CSS Properties

Here are some frequently used CSS properties:

- **Font Style**

```css
em {
  font-style: normal;
  color: red;
}
```

- **Font Weight**

```css
em {
  font-weight: bold;
}
```

- **Text Alignment**

```css
p {
  text-align: justify;
}
```

- **Colors**

```css
h1 {
  color: rebeccapurple;
```

```
}
h2 {
  color: crimson;
}
```

- **Background Colors**

```
strong {
  color: white;
  background-color: darkred;
}
```

- **Fonts**

```
body {
  font-family: "Helvetica", "Arial", sans-serif;
}
code {
  font-family: "Lucida Console", "Monaco", monospace;
}
```

- **Borders**

```
h1 {
  border: dashed thin red;
}
```

5. The CSS Box Model

Every element on a web page is treated as a **box**. Each box has:

- **Content** (text or image)
- **Padding** (space inside the border)
- **Border**
- **Margin** (space outside the border)

Example:

```
blockquote {
  border: thin solid grey;
  background-color: silver;
  padding: 0.25em;
  margin: 1em 2em;
}
```

The `box-sizing` property can make width/height calculations more intuitive by including padding and border.

6. CSS Units

CSS supports multiple units:

- **Absolute units**: `cm`, `mm`, `in`, `pt`
- **Pixels (`px`)**: tied to screen pixels
- **Viewport units**: `vh` (1% viewport height), `vw` (1% viewport width)
- **Relative units**:
  - `em` = current element's font size
  - `rem` = root font size (usually `<html>`)

Use `em`/`rem` for scalable designs, `px` for images, and `vw`/`vh` for layouts relative to screen size.

7. CSS Selectors

Selectors define **which elements** to style:

- **Tag selector**: `p { ... }`
- **Class selector**: `.optional { ... }`
- **ID selector**: `#first { ... }`

```
span.quantity {
  font-weight: bold;
}
.optional {
  color: grey;
}
```

- **Descendant selectors** (`h2 em`)
- **Child selectors** (`ul>li`)
- **Pseudo-classes** (dynamic states):

```
a:link {
  color: blue;
}
a:hover {
  color: darkblue;
}
```

- **Pseudo-elements** (parts of elements):

```
p::first-line {
  font-variant: small-caps;
}
```

8. Colors in CSS

Colors can be defined in different ways:

- **Named colors**: `red`, `crimson`

- **Hex codes**:
  - `#000` = black
  - `#fff` = white
  - `#f00` = red
- **RGB / HSL**:

```
rgb(178, 133, 224)
hsl(120, 100%, 40%)
```

Try experimenting with online color pickers like Google Color Picker.

9. Browser Compatibility

- Always use **valid HTML and CSS**.
- Not all browsers support the latest features equally.
- Check compatibility on W3Schools or MDN Web Docs.
- Consider reset.css or normalize.css for consistent cross-browser defaults.

10. Development Tools

Modern browsers provide **Developer Tools** (F12):

- Inspect HTML structure
- View applied CSS rules
- Experiment with changes in real time
- Visualize box models
- Debug network performance

Also consider using **Emmet** in your text editor to speed up coding:

```
li.optional: <li class="optional"></li>
ul>li: <ul><li></li></ul>
```

11. CSS Layout

Float and Clear

- `float: left/right`: pushes element to one side
- `clear: both`: prevents floats beside an element

Display Property

- **block**: takes full width (e.g., `<div>`)
- **inline**: flows in text (e.g., `<span>`)
- **inline-block**: inline but allows width/height
- **none**: hides element entirely

```
li {
  display: inline;
}
```

12. Summary: HTML + CSS

- **HTML**: defines structure and meaning.
- **CSS**: defines appearance.
- Keep them separate for flexibility.
- Add **classes/IDs** for fine control.
- Test across browsers and devices.

Next step: **JavaScript** for interactivity.