A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

For Floris:

Theory 1:

An HTML file is created similarly to a text file in Notepad. You type the content and save the file, but instead of using the .txt extension, you use .html or .htm. Sometimes, file extensions are not immediately visible in Windows Explorer. You can check the extension by right-clicking the file and selecting ‘Properties.’

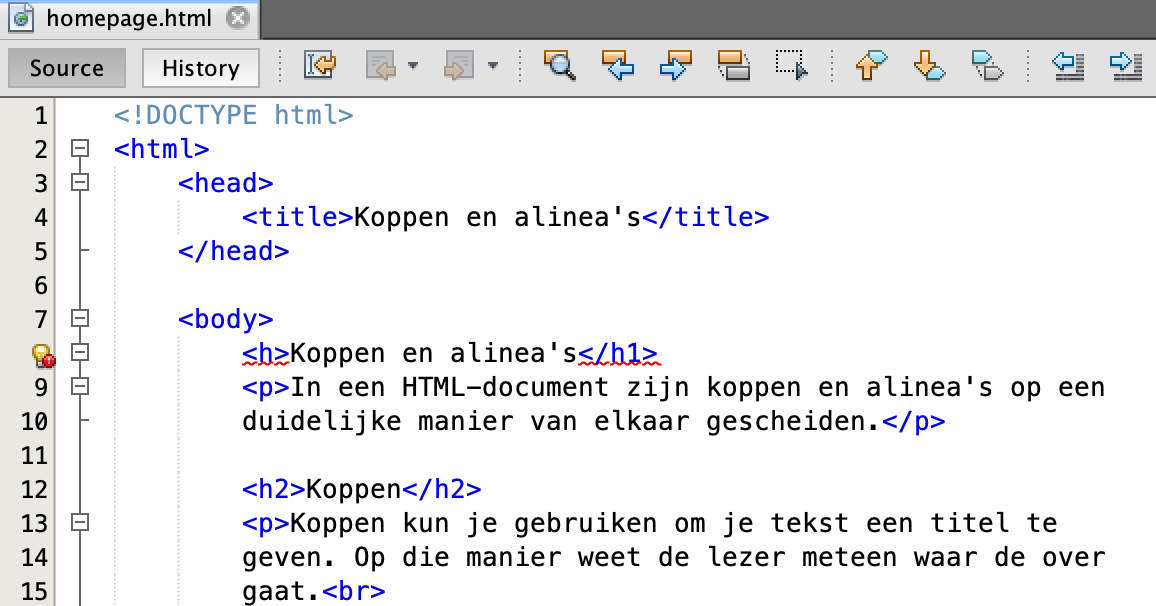
Although you can write HTML code in Notepad, there are specialized editors that make the process easier. Well-known examples include Notepad++, Visual Studio Code, and Netbeans. These programs offer additional functionalities such as syntax highlighting and error detection, making HTML coding more efficient.

Creating a new HTML page varies by editor, but usually, you start by creating a new file and saving it with the .html extension. It is important to choose a simple filename, preferably in lowercase and without spaces or special characters, such as assignment1.html or homepage.html. This helps prevent issues since some servers interpret spaces and special characters differently, for example, a space as %20. Additionally, servers are often case-sensitive, meaning filenames must match exactly.

To view an HTML page in a web browser, simply double-click the saved HTML document in Windows Explorer. The page will automatically open in your default browser.

Errors can occur quickly when coding in HTML. A small mistake can cause the page to display incorrectly. Therefore, it is important to check and validate your code. Some editors, such as Netbeans, highlight errors immediately. You can also use online validators, such as the one from W3C. Here, you can paste your HTML code or upload a file to detect errors.

Keep in mind that validation programs do not always fix all errors. Sometimes, your code may be technically correct but not structured as intended. That is why it is always wise to manually review your code.



Theory 2:

An HTML5 file does not just contain text; it also describes objects such as hyperlinks and images. This is done using elements. An HTML5 file is entirely composed of elements, meaning that all parts of a webpage belong to a specific element.

One example is the <title> element, which defines the name of the webpage. Each element usually consists of three parts:

* A **start tag**, such as <title>
* The **content**, for example, the page title
* An **end tag**, such as </title>

Most elements have both a start and an end tag, but some elements without content only have a single tag. The content of an element must always be placed between the start and end tag. Additionally, it is important to write all element names in lowercase.

### DOCTYPE Declaration

Every HTML file starts with a **DOCTYPE declaration**. This tells the web browser which version of HTML is being used. For HTML5, it is:

<!DOCTYPE html>

Although this declaration simply says "html" and does not explicitly mention "html5," it specifically applies to HTML5. In HTML4, the DOCTYPE declaration was much longer and more complex:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "https://www.w3.org/TR/html4/strict.dtd">

### The <html> Element

After the DOCTYPE declaration, the <html> element follows. This is the main element that contains all the HTML code for the webpage. An HTML5 file always includes exactly one <html> element, which houses all other elements. The basic structure looks like this:

<!DOCTYPE html>

<html>

</html>

To specify the language of the webpage, you can extend the <html> element with a language attribute:

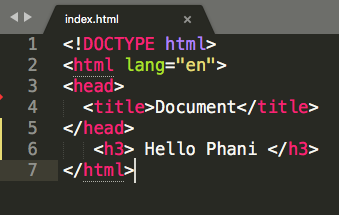
<html lang="en">

This helps search engines and web browsers correctly interpret the language of the page.

Inside the <html> element, two essential elements must always be present:

1. **The <head> element**: Contains metadata about the webpage, such as the title and links to styles or scripts.
2. **The <body> element**: Contains the actual content visible to the user, such as text, images, and links.

All other HTML elements must be placed inside either the <head> or <body> element.



Theory 3:

### **Headings in HTML**

In HTML, you can use headings and subheadings to indicate titles and subtitles within your text. This works similarly to how headings function in Microsoft Word. The largest heading is inserted using the <h1> element. Web browsers automatically adjust the text size, making the heading bold and placing it on a new line.

If you need a smaller heading, you can use <h2>, <h3>, <h4>, <h5>, or <h6>. There are six levels of headings in total, with <h1> being the largest and <h6> the smallest. The browser automatically reduces the text size for each successive heading level. Headings are important not only for structuring your webpage but also for search engine optimization (SEO), as they help define the content hierarchy.

### **Paragraphs in HTML**

Like headings, paragraph text must be placed within an element. In HTML, a paragraph is created using the <p> element. Web browsers automatically add spacing between paragraphs, making the text easier to read.

For example:

<p>This is a paragraph of text.</p>

<p>This is another paragraph.</p>

If you add multiple spaces or extra line breaks inside a paragraph element, the browser will not display them. This happens because web browsers treat extra spaces as a single space and ignore additional line breaks.

If you want to manually create a line break without starting a new paragraph, you should use the <br> element. This element does not need an end tag and forces text to move to the next line.

Example:

<p>This is a line of text.<br>This text appears on a new line.</p>

### **Text Formatting in HTML**

Although CSS is used for most website styling, HTML provides some basic formatting options for text. Below are a few elements used to format text:

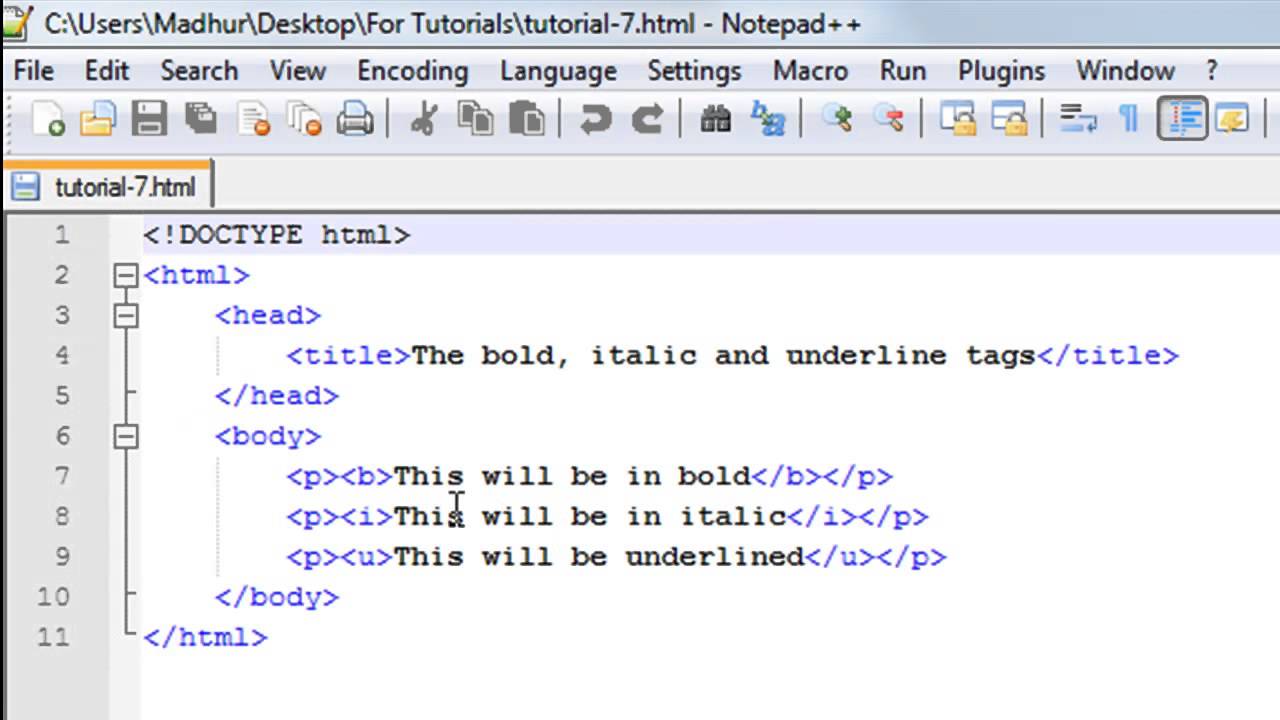
* **Bold text:** <b>Bold</b> → **Bold**
* **Italic text:** <i>Italic</i> → Italic
* **Underlined text:** <u>Underlined</u> → <u>Underlined</u>
* **Emphasized text:** <em>Emphasized</em> → Emphasized (for screen readers and SEO)

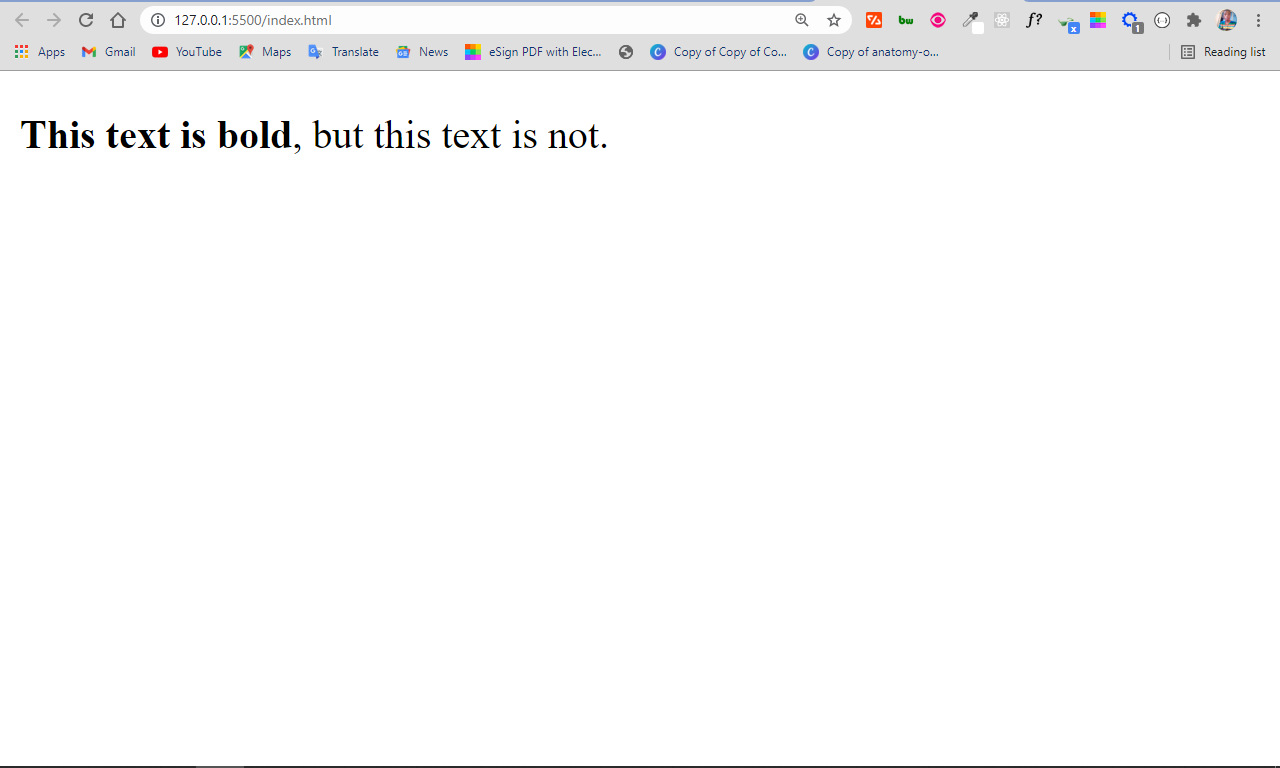
### **Special Characters in HTML**

If you want to display certain characters, such as extra spaces or symbols, you need to use HTML entities. Below are some commonly used special characters:

| **Code** | **Result** | **Name** |
| --- | --- | --- |
| &nbsp; |  | Non-breaking space |
| &quot; | " | Double quotation marks |
| &amp; | & | Ampersand |
| &lt; | < | Less than |
| &gt; | > | Greater than |
| &copy; | © | Copyright symbol |
| &euro; | € | Euro symbol |

These elements and characters help structure your webpage and ensure proper formatting across different browsers and devices.



For Kiara:

Theory 1:

### **Hyperlinks in HTML**

In printed books, references to other books require physically locating the correct page, which can be time-consuming. However, on the internet, this process is much simpler and faster using **hyperlinks**. A hyperlink allows users to click on a piece of text or an image to navigate to another webpage, document, or email.

There are three main types of hyperlinks:

1. **Fixed URL (Absolute URL)** – Links to an external website (e.g., https://www.google.com).
2. **Relative URL** – Links to a subpage within the same website (e.g., contact.html).
3. **Mailto URL** – Links to an email address and opens the user’s email client (e.g., mailto:someone@example.com).

### **Hyperlink with a Fixed URL**

A fixed URL points directly to an external website. Here’s an example of how to create a hyperlink to Google:

<!DOCTYPE html>

<html>

<head>

<title>Example of a hyperlink</title>

</head>

<body>

<p>A well-known website is <a href="https://www.google.com">the search engine Google</a>.</p>

</body>

</html>

In the code above, the phrase **"the search engine Google"** is enclosed within <a> tags, making it a clickable hyperlink. By default, web browsers style hyperlinks with blue text and underlining.

The key part of the hyperlink is the **href attribute** inside the <a> tag:

<a href="https://www.google.com">Google</a>

* The **href attribute** (Hypertext Reference) specifies the URL to which the hyperlink points.
* The value of the **href** attribute must be enclosed in quotation marks.
* It’s essential to include https:// in fixed URLs; otherwise, the browser may treat them as **relative URLs** instead.

### **Hyperlink with a Relative URL**

Unlike a fixed URL, a **relative URL** links to another page within the same website. Below is an example:

<!DOCTYPE html>

<html>

<head>

<title>Relative URL</title>

</head>

<body>

Click <a href="hyperlink.html">here</a> for the previous page.

</body>

</html>

In this example, the hyperlink points to hyperlink.html, but **without a full domain name**. This means that the browser will look for the file in the same directory as the current page.

#### **Linking to Files in Subfolders**

If a webpage is stored in a subfolder, you need to specify its location relative to the current file.

Example:

<a href="pages/contact.html">Contact</a>

This tells the browser to look inside the pages folder for contact.html.

Relative URLs are particularly useful when developing websites because they allow internal links to remain functional even if the website is moved to a different domain.

### **Hyperlink with a Mailto URL**

In addition to linking to webpages, hyperlinks can also create email links using the **mailto URL**. When clicked, a mailto link opens the user’s default email application with a pre-filled recipient address.

Example:

<a href="mailto:someone@example.com">Email me</a>

When a user clicks this link, their email client (e.g., Microsoft Outlook, Gmail) will open with a new email draft addressed to someone@example.com.

Theory 2:

### **Images in HTML**

To use images on a webpage, you use the **img element**. This element has no closing tag and is a **self-closing element**. All information about the image is stored within the **attributes** inside the opening tag.

Here is an example of using an image in HTML:

<!DOCTYPE html>

<html>

<head>

<title>An Image</title>

</head>

<body>

<p>

Here you can see my house.

<img src="house1.gif" alt="Roger's house." title="This is my house" width="150" height="188">

It is a house made of pixels, a very strong material.

</p>

</body>

</html>

### **Important Attributes of the img Element**

1. **src (source)**
   * The src value defines the URL or file name of the image.
   * If only a file name is given, like house1.gif, the browser treats it as a **relative URL**, meaning it assumes the image is in the same folder as the HTML file.
2. **alt (alternative text)**
   * The alt attribute provides a text description of the image in case the browser cannot load it.
   * This is **required** for accessibility and search engine optimization (SEO).
3. **title (tooltip)**
   * This attribute displays additional information when the user hovers over the image with the mouse.
   * It does not work on touchscreens.
4. **height and width**
   * These attributes define the height and width of the image in **pixels**.
   * You can specify only one (either height or width), and the browser will scale the image proportionally.
   * Incorrect dimensions can distort the image or disrupt the webpage layout.

Example of resizing an image:

<img src="house1.gif" alt="Roger's house." width="75">

If no height is specified and the image fails to load, the page layout may shift since the browser doesn’t reserve space for the missing image.

### **Using Images as Hyperlinks**

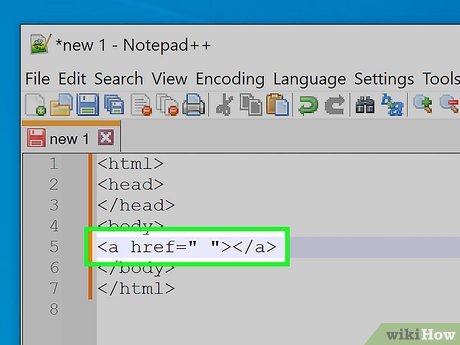
Sometimes, it is useful to make an image a **hyperlink** instead of text. This is commonly done with logos, which often link back to the homepage. You can achieve this by **nesting the img element inside an a element**:

<a href="https://www.google.com/images?q=house">

<img src="house1.gif" alt="Roger's house." title="This is my house" width="150" height="188">

</a>

When the user clicks on the image, the specified URL opens. In this example, the link leads to a Google image search for "house."



Theory 3:

### **Lists and Tables in HTML**

HTML provides different ways to structure and display lists and tables. You can create three types of lists:

1. **Unordered Lists (ul)**
   * Uses bullet points to list items.
   * Each item is enclosed in the <li> (list item) element.
   * Example:

<ul>

<li>Butter</li>

<li>Cheese</li>

<li>Eggs</li>

</ul>

* + This will display as:
    - Butter
    - Cheese
    - Eggs

1. **Ordered Lists (ol)**
   * Uses numbers instead of bullet points.
   * The structure is the same as unordered lists but with the <ol> element.
   * Example:

<ol>

<li>Butter</li>

<li>Cheese</li>

<li>Eggs</li>

</ol>

* + This will display as:
    - Butter
    - Cheese
    - Eggs



1. **Definition Lists (dl)**
   * Used to define terms and their descriptions.
   * Uses <dl> (definition list), <dt> (definition term), and <dd> (definition description).
   * Example:

<dl>

<dt>Butter</dt>

<dd>Churned cream from milk.</dd>

<dt>Cheese</dt>

<dd>Also made from milk.</dd>

<dt>Eggs</dt>

<dd>Eggs come from birds, not milk.</dd>

</dl>

* + This will display as:

**Butter**  
Churned cream from milk.

**Cheese**  
Also made from milk.

**Eggs**  
Eggs come from birds, not milk.

### **Tables in HTML**

Tables are created using the <table> element. They consist of rows (<tr>) and cells (<td>).

**Basic Table Structure:**

<table>

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

</tr>

<tr>

<td>Cell 3</td>

<td>Cell 4</td>

</tr>

</table>

This creates a simple table:

| **Cell 1** | **Cell 2** |
| --- | --- |
| Cell 3 | Cell 4 |

By default, tables have no borders. To add a border, use the border attribute:

<table border="1">

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

</tr>

<tr>

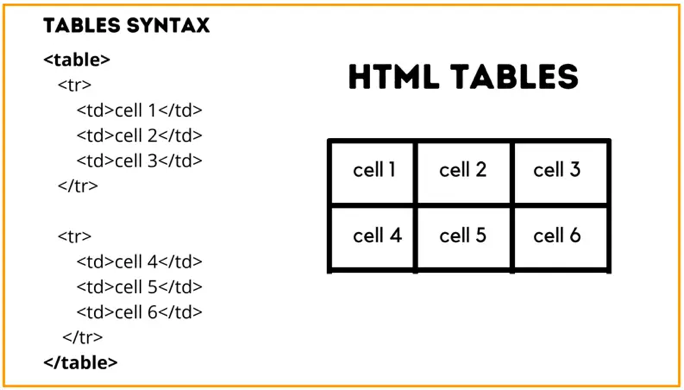
<td>Cell 3</td>

<td>Cell 4</td>

</tr>

</table>

This adds a 1-pixel border around the table.



### **Merging Cells**

HTML allows you to merge multiple cells using the **colspan** (for columns) and **rowspan** (for rows) attributes.

#### **Merging Columns (**colspan**)**

To merge two columns:

<table border="1">

<tr>

<td colspan="2">Cell 1 & Cell 2</td>

</tr>

<tr>

<td>Cell 3</td>

<td>Cell 4</td>

</tr>

</table>

This makes "Cell 1 & Cell 2" span across two columns.

| **Cell 1 & Cell 2** |
| --- |
| Cell 3 |

#### **Merging Rows (**rowspan**)**

To merge two rows:

<table border="1">

<tr>

<td rowspan="2">Cell 1 & Cell 3</td>

<td>Cell 2</td>

</tr>

<tr>

<td>Cell 4</td>

</tr>

</table>

This makes "Cell 1 & Cell 3" span across two rows.

| **Cell 1 & Cell 3** | **Cell 2** |
| --- | --- |
|  | Cell 4 |

For me:

Theory 1:

### **Embedding Multimedia from Third-Party Websites**

When building a website, you may want to display multimedia content from external sources, also known as **third-party content**. A common example is embedding a **YouTube video** directly into your webpage. To achieve this, you use the **iframe (inline frame) element**, which allows you to embed an external website or multimedia content within your own site.

#### **Using an Iframe to Embed a Website**

An iframe enables you to display an entire website within another webpage. The basic syntax looks like this:

<iframe width="560" height="315" src="https://fundament-online.nl/"></iframe>

However, some websites **block iframe embedding** for security reasons. If a website does not support iframe embedding, it **will not be displayed** inside the frame.

#### **Embedding YouTube Videos in HTML**

Embedding **YouTube videos** is one of the most useful applications of iframes. YouTube makes this process easy by providing an **embedding feature**.



To embed a YouTube video:

1. **Go to the YouTube video** you want to add to your site.
2. **Click on “Share”** under the video.
3. **Select the “Embed” option.**
4. Copy the provided <iframe> code and paste it into your HTML file.

The embedded code typically looks like this:

<iframe width="560" height="315"

src="https://www.youtube.com/embed/LMd8rG2WDbc"

frameborder="0"

allow="accelerometer; autoplay; encrypted-media; gyroscope; picture-in-picture"

allowfullscreen>

</iframe>

### **Breaking Down the Code**

* **src Attribute**: The src (source) contains a **special embed URL** provided by YouTube. Notice that instead of the normal YouTube video URL, the embed URL includes **"/embed/"**.
  + You **cannot** simply copy the URL from the browser’s address bar; you must use the embed code that YouTube provides.
* **width and height Attributes**: These define the **size of the iframe**. You can adjust them to fit your layout.
* **frameborder="0"**: This removes the default border around the iframe. If omitted, a thin border might appear.
* **allowfullscreen**: This enables fullscreen mode, allowing users to expand the video to full screen.
* **Additional allow Attributes**:
  + accelerometer: Enables motion-based controls.
  + autoplay: Allows the video to play automatically (some browsers may block this).
  + encrypted-media: Allows playback of encrypted content.
  + gyroscope: Enables gyroscope-related functionality (often used in VR videos).
  + picture-in-picture: Allows the video to be viewed in a small floating window.

A screenshot of a video chat

Description automatically generated

### **Customization Options**

You can customize the appearance and behavior of the embedded video:

1. **Change Video Size**
   * Adjust the width and height attributes. Example:

<iframe width="800" height="450" src="https://www.youtube.com/embed/LMd8rG2WDbc"></iframe>

1. **Autoplay the Video**
   * Add ?autoplay=1 to the URL. Example:

<iframe width="560" height="315" src="https://www.youtube.com/embed/LMd8rG2WDbc?autoplay=1"></iframe>

* + Note: Some browsers **block autoplay** unless the video is muted (mute=1).

1. **Start the Video at a Specific Time**
   * Add ?start=30 to start at 30 seconds. Example:

<iframe width="560" height="315" src="https://www.youtube.com/embed/LMd8rG2WDbc?start=30"></iframe>

1. **Hide Video Controls**
   * Add ?controls=0 to hide the play/pause buttons. Example:

<iframe width="560" height="315" src="https://www.youtube.com/embed/LMd8rG2WDbc?controls=0"></iframe>

### **When Not to Use Iframes**

While iframes are useful, they also have **drawbacks**:

* Some websites **block embedding** for security reasons.
* They can **slow down your page load speed** if too many are used.
* **SEO Issues**: Content inside an iframe is **not indexed** by search engines like Google.

Theory 2:

**CSS and Its Use with HTML**

With HTML5, there is a clear separation between content (HTML pages) and design. The styling is done in a separate file known as a **CSS file**. CSS stands for **"Cascading Style Sheet”.**

The CSS file contains all the styling rules for the website and applies them to all HTML pages. This solves the problem of having to manually update the style on multiple pages—if you need to change the background color, you only need to update it in one place (the CSS file), and it will be applied to all pages.

The CSS file is saved with the **.css** file extension, and it is common to name it **"style.css"**.



**Linking CSS to HTML**

To link a CSS file to an HTML file, you must include it inside the <head> element of the HTML document. The <head> section contains information about the website that is not directly visible on the page.

To link the CSS file, add the following line inside the <head> section:

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

The updated <head> section now looks like this:

<head>

<title>Working with CSS</title>

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

</head>

This <link> element has three attributes:

1. **rel** – Specifies the relationship between the HTML document and the linked file (in this case, a stylesheet).
2. **type** – Specifies the type of file being linked (always "text/css" for stylesheets).
3. **href** – Defines the location and name of the CSS file.

**Creating a CSS File**

A CSS file is created similarly to an HTML file but is saved with the **.css** extension. It is common to name it **style.css**.

Many code editors allow you to view multiple files side by side, so you can see both the HTML and CSS code at the same time.

CSS uses **selectors** to apply styles to HTML elements. The most common ones are:

* **Tag selectors** (apply styles to all elements of a given type)
* **ID selectors** (apply styles to specific elements with an ID)

**Tag Selector Example**

For example, an <h1> heading in HTML is usually black by default. To change its text color to red using CSS, add this code to **style.css**:

h1 {

color: red;

}

**CSS Syntax**

The structure of a CSS rule follows this format:

selector {

property: value;

}

* The **selector** defines which HTML element(s) the rule applies to.
* The **property** specifies the styling feature (e.g., color, font-size).
* The **value** defines the style to apply.
* Each rule ends with a **semicolon (;)**.

**Changing Text Color in CSS**

You can define text color in three ways:

1. **Keywords** – CSS has 140 predefined color names, such as red, blue, gray, gold, etc.
2. **RGB values** – Define colors using Red, Green, and Blue values (0 to 255 each):

h1 {

color: rgb(0, 0, 255); /\* Blue \*/

}

1. **Hexadecimal values** – Represent colors in a shorter format, starting with #:

h1 {

color: #e07ae6; /\* Purple \*/

}



**Setting a Background in CSS**

You can change the background of a website in two ways:

1. **Using a solid color:**

body {

background-color: lightblue;

}

1. **Using an image as the background:**

body {

background-image: url('image.png');

}

If the image is too small, the browser will tile it across the screen. Be mindful that different devices (smartphones, 4K monitors) have different screen sizes, so background images may not always display as intended.

**Changing Fonts in CSS**

To change the font of a website, use the **font-family** property:

body {

font-family: Verdana;

}

It is recommended to specify multiple font options in case the user’s device does not have the primary font installed:

body {

font-family: Rockwell, Arial;

}

In this example, if Rockwell is unavailable, Arial will be used instead.



**Adjusting Text Size**

Browsers set default sizes for text, but you can override them using the **font-size** property:

h1 {

font-size: 32pt;

}

Font size can be specified in different units:

* **Points (pt)** – Common in Word documents (e.g., 10pt–12pt for normal text).
* **Pixels (px)** – Fixed-size text.
* **Percentages (%)** – Relative to the default size.
* **Viewport width (vw)** – Adapts to screen size.
* **Em units (em)** – Relative to the parent element’s font size (e.g., 2em is twice the default size).

You can also set multiple styles at once:

h1 {

font-family: Arial;

font-size: 24pt;

color: red;

}

**Text Alignment in CSS**

By default, text is left-aligned. You can change this using the **text-align** property:

h1 {

text-align: center;

}

p {

text-align: right;

}

Available options:

* left (default)
* right
* center
* justify (stretches text to align with both left and right margins)

**Text Formatting (Bold, Italic, Underline)**

CSS provides three properties for text styling:

* **Bold:** font-weight: bold;
* **Italic:** font-style: italic;
* **Underline:** text-decoration: underline;

Example:

h1 {

font-weight: normal; /\* Removes bold from h1 \*/

}

p {

font-weight: bold; /\* Makes paragraphs bold \*/

}

For underlining, striking through, or adding a line above the text:

h1 {

text-decoration: line-through; /\* Strikethrough \*/

}

h2 {

text-decoration: overline; /\* Line above \*/

}

p {

text-decoration: underline; /\* Underline \*/

}

Theory 3:

### **Styling Hyperlinks with CSS**

* The <a> element is used to style hyperlinks.
* Example:

a {

color: blue;

}

* The :hover pseudo-class is used to change the appearance when a user hovers over a link.

a:hover {

color: darkblue;

text-decoration: none;

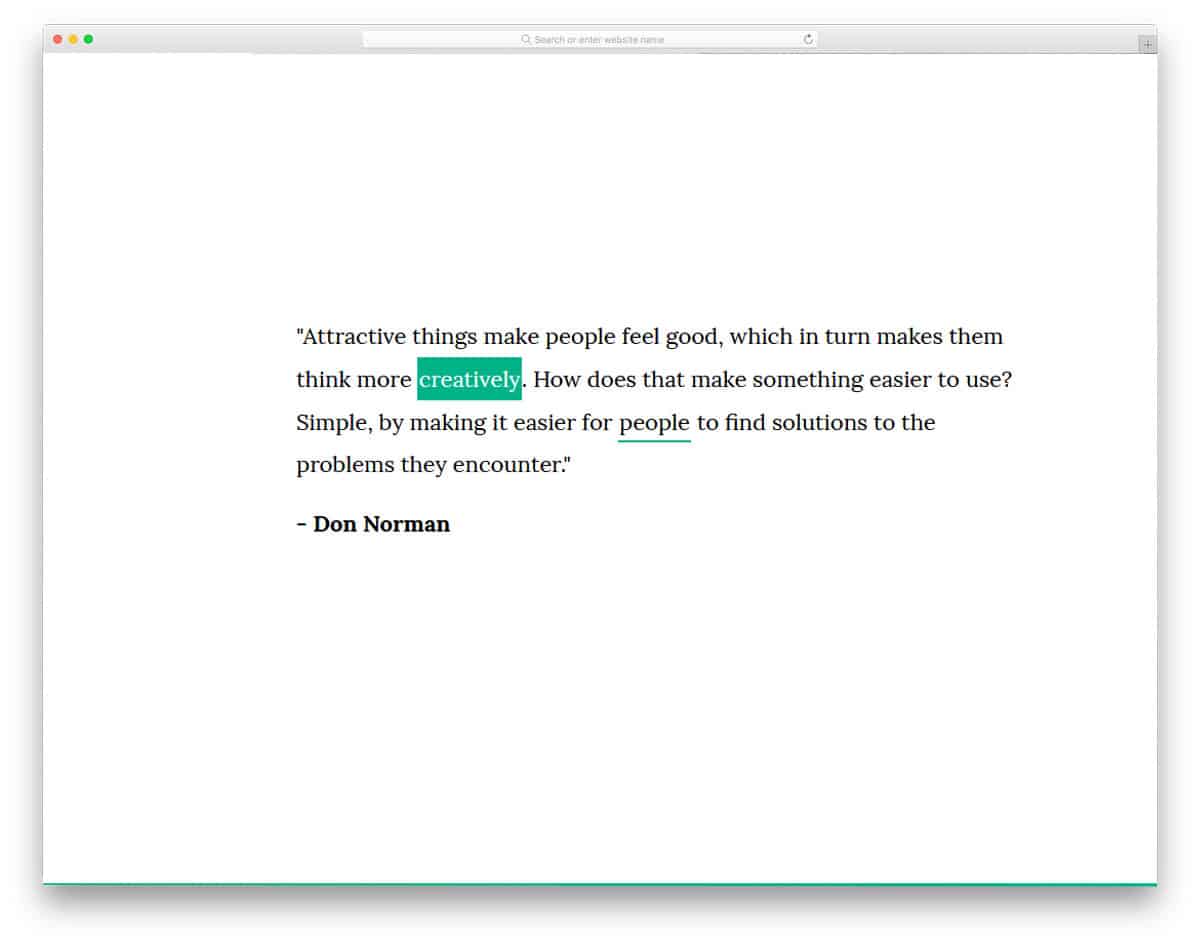
}

* The **cursor** property allows customization of the mouse pointer when hovering over a link, such as changing it to a help cursor:

a:hover {

cursor: help;

}



### **Lists in CSS**

There are **three types of lists** in HTML:

1. **Unordered lists (<ul>)** – Use bullet points (disc, square, etc.).
2. **Ordered lists (<ol>)** – Use numbers or letters (1, A, I, etc.).
3. **Definition lists (<dl>)** – Used for term-definition pairs.

CSS allows customization of list markers with list-style-type:

* Example for an **unordered list** with square bullets:

ul {

list-style-type: square;

color: blue;

}

* Example for an **ordered list** with Roman numerals:

ol {

list-style-type: upper-roman;

}

### **Styling Tables with CSS**

* By default, **HTML tables have no borders**, but they can be styled using CSS.
* The border property allows customization of:
  1. **Thickness** (in pixels)
  2. **Border type** (solid, dotted, dashed, etc.)
  3. **Color**

table {

border: 1px solid black;

}

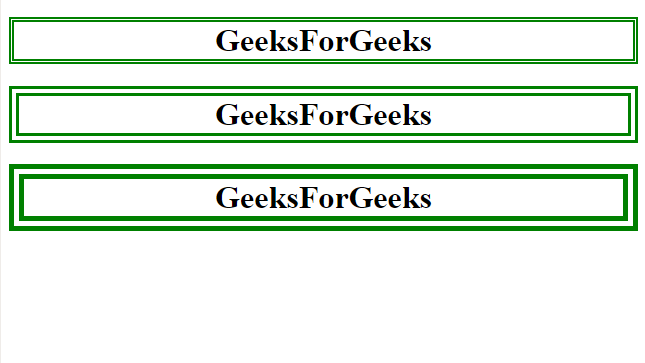
* To apply the same border to **table cells (<td>)**, use:

td {

border: 1px solid black;

}

However, this creates a **double border** because both the table and cells have their own borders.



* To remove the double border, use border-collapse:

table {

border: 1px solid black;

border-collapse: collapse;

}

* By default, the **table width adjusts based on content**.
* You can manually set the width using pixels or percentages:

table {

width: 250px;

}

### **Grouping Multiple HTML Elements in CSS**

* Instead of writing separate CSS rules for similar elements, you can group them:

h1, h2, h3, h4 {

color: green;

}

This applies the same style to **all selected elements** at once.