**Assignment-1**

1. **Explain tags of HTML 5?**

* **Tags of HTML 5**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| **<article>** | It defines the independent or self-contained content of a webpage. |
| **<aside>** | It defines the content which provide information about the main content. |
| **<bdi>** | It is used to isolate the part of text which might be formatted in another direction. |
| **<details>** | It defines additional information which only visible as per user demand. |
| **<dialog>** | It represents a dialog box or other interactive components. |
| **<figcaption>** | It defines caption for the <figure> element. |
| **<figure>** | It defines a self-contained content, and referenced as a single unit. |
| **<footer>** | It represents the footer section of the webpage. |
| **<header>** | It defines the introductory or navigational content of the webpage. |
| **<main>** | It specifies the main content of the HTML document. |
| **<mark>** | It represent the text which is highlighted or marked for reference or notation purposes. |
| **<meter>** | It represents a scalar value within a known range. |
| **<nav>** | It represents the section which contains navigation links. |
| **<progress>** | It defines a progress bar which shows completions progress of a task. |
| **<rp>** | It defines alternative content for the browser which do not support ruby annotations. |
| **<rt>** | It defines explanations and pronunciations of characters in ruby annotations. |
| **<ruby>** | It defines ruby annotations (Specifically for Asian language). |
| **<section>** | It defines a generic section within an HTML document. |
| **<summary>** | It defines summary or caption for a <details> element which can be clicked to change the state of <details> element. |
| **<time>** | It defines data/time within an HTML document. |
| **<wbr>** | It specifies a line break opportunity. (Where line break possible) |

1. **Explain all feature of HTML5 feature with example?**

* **HTML5 few advance features:** Semantic Elements, Audio and Video Support, Canvas Elements, Geolocation API, Local Storage, Responsive Images, Web Workers, Drag and Drop API, Form Enhancements, Web Sockets, Micro Data, Cross Document Messaging.
* **Semantic Elements:** HTML5 introduced semantic elements that provide a more meaningful way to structure web content. These include <header>, <nav>, <main>, <article>, <section>, <aside>, <footer>, and more. They help improve accessibility and SEO.

|  |
| --- |
| <header>  <h1>Website Header</h1>  </header>  <nav>  <ul>  <li><a href="#">Home</a></li>  <li><a href="#">About</a></li>  <li><a href="#">Contact</a></li>  </ul>  </nav>  <main>  <article>  <h2>Article Title</h2>  <p>Article content goes here.</p>  </article>  </main>  <footer>  <p>&copy; 2023 Your Website</p>  </footer> |

* **Audio and Video Support:** HTML5 introduced native support for embedding audio and video content using the <audio> and <video> elements.

|  |
| --- |
| <audio controls>  <source src="audio.mp3" type="audio/mpeg"> Your browser does not support the audio element.  </audio>  <video controls width="320" height="240">  <source src="video.mp4" type="video/mp4">Your browser does not support the video element.  </video> |

* **Canvas:** The <canvas> element allows for dynamic, scriptable rendering of 2D graphics. It's commonly used for games, data visualization, and interactive graphics.

|  |
| --- |
| <canvas id="myCanvas" width="400" height="200"></canvas>  <script>  var canvas = document.getElementById('myCanvas');  var ctx = canvas.getContext('2d');  ctx.fillStyle = 'red';  ctx.fillRect(50, 50, 100, 100);  </script> |

* **Geolocation:** HTML5 enables websites to access the user's geographical location through the Geolocation API.

|  |
| --- |
| <button onclick="getLocation()">Get Location</button>  <p id="demo"></p>  <script>  function getLocation() {  if (navigator.geolocation) {  navigator.geolocation.getCurrentPosition(showPosition);  } else {  document.getElementById('demo').innerHTML = 'Geolocation is not supported by this browser.';  }  }  function showPosition(position) {  document.getElementById('demo').innerHTML = 'Latitude: ' + position.coords.latitude + '<br>Longitude: ' + position.coords.longitude;  }  </script> |

* **Drag and Drop:** HTML5 enables easy implementation of drag-and-drop functionality for elements on a web page.

|  |
| --- |
| <div id="dragTarget" draggable="true">Drag me!</div>  <div id="dropTarget">Drop here!</div>  <script>  var dragTarget = document.getElementById('dragTarget');  var dropTarget = document.getElementById('dropTarget');  dragTarget.addEventListener('dragstart', function (event) {  event.dataTransfer.setData('text', event.target.id);  });  dropTarget.addEventListener('dragover', function (event) {  event.preventDefault();  });  dropTarget.addEventListener('drop', function (event) {  event.preventDefault();  var data = event.dataTransfer.getData('text');  event.target.innerText = 'Dropped: ' + data;  });  </script> |

1. **Describe audio & video support in HTML 5 with example?**

* **Audio and Video Support:** One of the many HTML 5 features is the support for audio and video. It has reduced the hassle of relying upon third-party services such as Adobe Flash player. To embed Audio and Video into your HTML document, you may use the following two tags, <audio> and <video> tags. These two tags are launched with the release of HTML 5 and support a numerous range of attributes such as height, width, and more that offers developers to leverage the customization of HTML documents.
* **Example:**

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <title>HTML5</title>  </head> <body>  <!-- Code to setup video -->  <video width = "300" height = "200" controls autoplay>  <source src = "./dog.mp4" type ="video/mp4" />  </video>  <!-- Code to setup audio -->  <audio controls>  <source src="dog.mp3" type="audio/mp3">  </audio> </body>  </html> |

1. **How to importing External Style Sheets in CSS3?**

* **Importing External Style Sheets in CSS:** At-rules are clever, powerful little huggers that encapsulate a bunch of CSS rules and apply them to something specific. They can be used to import other CSS files, apply CSS to a particular media, or embed funky uncommon fonts.
* @import rule allows you to import a style sheet into another style sheet.
* @import rule must be at the top of the document (but after any @charset declaration).
* @import rule also supports media queries, so you can allow the import to be media-dependent.
* **Syntax:** The syntax of @import rule is as follows, **import /\*url or list-of-media-queries\*/**

|  |  |
| --- | --- |
| **Value** | **Description** |
| **url|string** | A url or a string representing the location of the resource to import. The url may be absolute or relative |
| **list-of-mediaqueries** | A comma-separated list of media queries conditioning the application of the CSS rules defined in the linked URL |

* The media queries can be compound statements which may specify the behavior of the document in different media.
* **The following examples implement the @import rule:**

|  |
| --- |
| <!DOCTYPE html>  <html>  <head> <style type="text/css">  @import url(style.css);  body{  background-color: honeydew;  } </style>  </head> <body>  <p>This is demo paragraph one. </p>  <p class="two">This is demo paragraph two.</p>  <p>This is demo paragraph three</p>  </body>  </html> |

* **Import the “new-style.css" style sheet ONLY if the media is print:** @import "new-style.css" print;
* **Import the “latest-style.css" style sheet ONLY if the media is screen and the viewport is maximum 900 pixels:** @import "latest-style.css" screen and (max-width: 900px);

1. **Short note on new Features of CSS 3?**

* **Some New Features of CSS 3:**

1. **Selectors:** Selectors in CSS 3 are very interesting. They allow the designer/developer to select on much more specific levels of the document.some selectors are:
2. matching on attributes and attribute values, including partial matches
3. structural pseudo-classes, such as the nth-child
4. a target pseudo-class to style only elements that are targetted in the URL
5. a checked pseudo-class to style any element that is checked such as radio or checkbox elements
6. **Text Effects and Layout:** Making changes to the hyphenation, whitespace, and justification of text in documents.
7. **First-Letter and First-Line Pseudo-Classes:** CSS 3 should allow properties to affect the kerning and alignment of [drop-caps](https://www.thoughtco.com/css-initial-caps-3466212).
8. **Paged Media and Generated Content:** CSS 3 now supports more options in paged media, such as running headers, footers, and page numbers. Plus there will be advanced properties for printing generated content including properties for footnotes and cross-references.
9. **Multi-Column Layout:** Right now, the multi-column layout working draft provides properties to allow designers to display their content in multiple columns with definitions like the column-gap, column-count, and column-width.
10. **Ruby:** CSS will now support the ability to add small annotations on top or next to words, most often used in Chinese and Japanese. They are generally used to give the pronunciation or meaning of difficult ideograms.
11. **Describe @charset Rule in CSS 3 with syntax?**

* **CSS @charset Rule:**
* The @charset rule specifies the character encoding used in the style sheet. The @charset rule must be the first element in the style sheet and not be preceded by any character. If several @charset rules are defined, only the first one is used. The @charset rule cannot be used inside a style attribute (on an HTML element), or inside the <style> element where the character set of the HTML page is relevant.

|  |
| --- |
| * **Syntax:**   **@charset "utf-8";** |

1. **Short note on advance CSS 3 with Alpha transparency?**

* Alpha transparency in CSS3 is a way to control the opacity of an element. Opacity is a measure of how visible an element is, with a value of 1 being fully opaque and a value of 0 being fully transparent. Alpha transparency can be used to create a variety of effects, such as:
* Fading elements in and out
* Creating ghost buttons
* Overlaying elements on top of each other
* Creating semi-transparent backgrounds

**To apply alpha transparency to an element, you use the** opacity **property.** The opacity property accepts a value between 0 and 1, where 0 is fully transparent and 1 is fully opaque. For example, the following CSS code will set the opacity of the div element to 50%, making it semi-transparent:

|  |
| --- |
| **div {**  **opacity: 0.5;**  **}** |

* Alpha transparency can also be used to create RGBA color values. RGBA color values are an extension of RGB color values with an alpha channel, which specifies the opacity for a color. An RGBA color value is specified with: **rgba(red, green, blue, alpha)**
* The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque). For example, the following CSS code will set the background color of the div element to a semi-transparent green color:

|  |
| --- |
| div {  background-color: rgba(0, 255, 0, 0.5);  } |

* **Alpha transparency can be used to create a variety of advanced CSS effects. For example, you can use alpha transparency to:**
* Create a ghost button that is visible when the user hovers over it, but disappears when they move their cursor away.
* Overlay a semi-transparent image over a background image to create a fading effect.
* Create a semi-transparent background for a modal dialog to make it easier to see the underlying content.
* **Here are some examples of advanced CSS effects that can be created with alpha transparency:**

|  |
| --- |
| **/\* Ghost button \*/**  .ghost-button {  opacity: 0.5;  transition: opacity 0.2s;  }  .ghost-button:hover {  opacity: 1;  }  **/\* Fading image overlay \*/**  .image-overlay {  position: absolute;  top: 0;  left: 0;  width: 100%;  height: 100%;  background-image: url('https://example.com/image.jpg');  opacity: 0.5;  }  **/\* Modal dialog background \*/**  .modal-dialog {  position: fixed;  top: 0;  left: 0;  width: 100%;  height: 100%;  background-color: rgba(0, 0, 0, 0.5);  z-index: 100;  } |

1. **How to embed fonts in CSS 3?**

* In CSS3, We can embed fonts using the **@font-face** rule. This rule allows you to specify custom fonts that will be downloaded and used by the browser when rendering your web page. Here's how you can embed fonts in CSS3:

1. **Choose a Font:** First, you need to select the font you want to embed on your website. You can use web fonts from various sources, including Google Fonts, Adobe Fonts, or your own custom font files (e.g., .woff or .woff2 files).
2. **Download or Host the Font Files:** If you're using a web font service like Google Fonts or Adobe Fonts, they will provide you with a link to include in your HTML. If you have custom font files, make sure they are hosted on your web server.
3. **Define the @font-face Rule:** In your CSS file, use the @font-face rule to define the font family and specify the source of the font files. Here's an example:

|  |
| --- |
| @font-face {  font-family: 'YourFontName';  src: url('path/to/your-font.woff2') format('woff2'),  url('path/to/your-font.woff') format('woff');/\* Add additional font formats here if needed \*/  } |

* **font-family:** This is the name you'll use to reference the font in your CSS styles.
* **src:** This property specifies the source of the font files. You should provide the path to your font files and specify the format (e.g., woff, woff2).

1. **Use the Custom Font:** Once you've defined the **@font-face** rule, you can use your custom font in your CSS styles by specifying the font-family property:

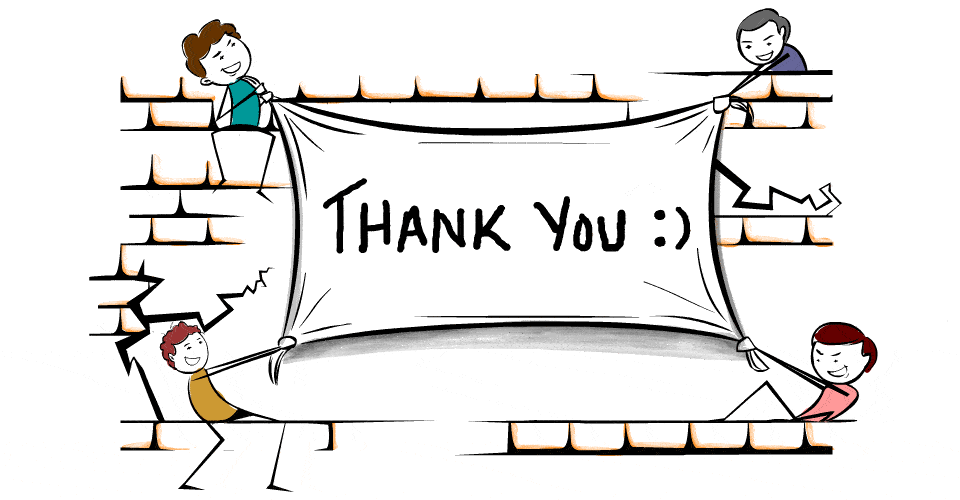
|  |
| --- |
| body {  font-family: 'YourFontName', sans-serif;  } |

In this example, 'YourFontName' should match the font-family name you used in the @font-face rule. If the custom font fails to load, the browser will fall back to a generic sans-serif font.

1. **Cross-Browser Compatibility:** To ensure cross-browser compatibility, you may want to include multiple font formats in your src property within the @font-face rule. Different browsers support different font formats, so having multiple formats (e.g., woff and woff2) can improve compatibility.
2. **Font License:** Make sure you have the proper licenses or permissions to use and embed the fonts on your website, especially if you're using custom fonts.
3. **Testing:** After embedding the fonts, thoroughly test your website in various browsers and devices to ensure that the fonts are loading correctly and rendering as expected.

By following these steps, you can embed custom fonts into your CSS3 styles, giving your website a unique and distinctive typographic style.

**(PPT Give you more understanding than PDF)** The material for the PDF has been compiled from various sources such as books, tutorials (offline and online), lecture notes, several resources available on Internet. The information contained in this PDF is for general information and education purpose only. While we endeavor to keep the information up to date and correct, **we make no representation of any kind about the completeness and accuracy of the material.** The information shared through this PDF material should be used for educational purpose only.

****