CALIBRAINT TECHNOLOGIES

MILESTONE 5 - 06.09.2024
Selectors and Combinators
Pseudo Classes and Elements
!important
CSS Functions
Comments

Selectors and Combinators

- CSS selectors are used to "find" (or select) the HTML elements you want to style.
- We can divide CSS selectors into five categories:
    - 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)
- **ELEMENT SELECTOR**
    - The element selector selects HTML elements based on the element name.
- **ID SELECTOR**
    - The id selector uses the id attribute of an HTML element to select a specific element.
    - The id of an element is unique within a page, so the id selector is used to select one unique element!
    - To select an element with a specific id, write a hash (#) character, followed by the id of the element.
- **CLASS SELECTOR**
    - The class selector selects HTML elements with a specific class attribute.
    - To select elements with a specific class, write a period (.) character, followed by the class name.
    - A class name cannot start with a number.
- **UNIVERSAL SELECTOR**
    - The universal selector (*) selects all HTML elements on the page.
- **GROUPING SELECTOR**
    - The grouping selector selects all the HTML elements with the same style definitions.

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

COMBINATORS

- A combinator is something that explains the relationship between the selectors.
- A CSS selector can contain more than one simple selector. Between the simple selectors, we can include a combinator.
- There are four different combinators in CSS:
    - descendant selector (space)
    - child selector (>)
    - adjacent sibling selector (+)
    - general sibling selector (~)
- **Descendant Selector**
    - The descendant selector matches all elements that are descendants of a specified element.
- **Child Selector (>)**
    - The child selector selects all elements that are the children of a specified element.
- **Adjacent Sibling Selector (+)**
    - The adjacent sibling selector is used to select an element that is directly after another specific element.
    - Sibling elements must have the same parent element, and "adjacent" means "immediately following".
- **General Sibling Selector (~)**
    - The general sibling selector selects all elements that are next siblings of a specified element.

All CSS Combinator Selectors

Selector	Example	Example description
element element	div p	Selects all <p> elements inside elements
element>element	div > p	Selects all <p> elements where the parent is a element
element+element	div + p	Selects the first <p> element that are placed immediately after elements
element1~element2	p ~ ul	Selects every element that are preceded by a <p> element

Pseudo Classes and Elements

- **PSEUDO CLASS**
- pseudo-class is used to define a special state of an element.
    - Style an element when a user mouses over it
    - Style visited and unvisited links differently
    - Style an element when it gets focus
- Syntax :

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

Selector	Example	Example description
:active	a:active	Selects the active link
:checked	input:checked	Selects every checked <input> element
:disabled	input:disabled	Selects every disabled <input> element
:empty	p:empty	Selects every <p> element that has no children
:enabled	input:enabled	Selects every enabled <input> element
:first-child	p:first-child	Selects every <p> elements that is the first child of its parent
:first-of-type	p:first-of-type	Selects every <p> element that is the first <p> element of its parent
:focus	input:focus	Selects the <input> element that has focus
:hover	a:hover	Selects links on mouse over
:in-range	input:in-range	Selects <input> elements with a value within a specified range
:invalid	input:invalid	Selects all <input> elements with an invalid value
:lang(*language*)	p:lang(it)	Selects every <p> element with a lang attribute value starting with "it"
:last-child	p:last-child	Selects every <p> elements that is the last child of its parent
:last-of-type	p:last-of-type	Selects every <p> element that is the last <p> element of its parent
:link	a:link	Selects all unvisited links
:not(selector)	:not(p)	Selects every element that is not a <p> element
:nth-child(n)	p:nth-child(2)	Selects every <p> element that is the second child of its parent

:nth-last-child(n)	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child
:nth-last-of-type(n)	p:nth-last-of-type(2)	Selects every <p> element that is the second <p> element of its parent, counting from the last child
:nth-of-type(n)	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent
:only-of-type	p:only-of-type	Selects every <p> element that is the only <p> element of its parent
:only-child	p:only-child	Selects every <p> element that is the only child of its parent
:optional	input:optional	Selects <input> elements with no "required" attribute
:out-of-range	input:out-of-range	Selects <input> elements with a value outside a specified range
:read-only	input:read-only	Selects <input> elements with a "readonly" attribute specified
:read-write	input:read-write	Selects <input> elements with no "readonly" attribute
:required	input:required	Selects <input> elements with a "required" attribute specified
:root	root	Selects the document's root element
:target	#news:target	Selects the current active #news element (clicked on a URL containing that anchor name)
:valid	input:valid	Selects all <input> elements with a valid value
:visited	a:visited	Selects all visited links

- **PSEUDO ELEMENTS**
- A CSS pseudo-element is used to style specified parts of an element.
    - Style the first letter, or line, of an element
    - Insert content before, or after, the content of an element
- Syntax

  selector::pseudo-element {

  property: value;}

- **The ::first-line Pseudo-element**
    - The ::first-line pseudo-element is used to add a special style to the first line of a text.
    - The ::first-line pseudo-element can only be applied to block-level elements.The following properties apply to the ::first-line pseudo-element:
        - font properties
        - color properties
        - background properties
        - word-spacing
        - letter-spacing
        - text-decoration
        - vertical-align
        - text-transform
        - line-height
        - clear
- Notice the double colon notation - ::first-line versus :first-line.

- The double colon replaced the single-colon notation for pseudo-elements in CSS3. This was an attempt from W3C to distinguish between pseudo-classes and pseudo-elements.
- The single-colon syntax was used for both pseudo-classes and pseudo-elements in CSS2 and CSS1.
- For backward compatibility, the single-colon syntax is acceptable for CSS2 and CSS1 pseudo-elements.

Selector	Example	Example description
::after	p::after	Insert something after the content of each <p> element
::before	p::before	Insert something before the content of each <p> element
::first-letter	p::first-letter	Selects the first letter of each <p> element
::first-line	p::first-line	Selects the first line of each <p> element
::marker	::marker	Selects the markers of list items
::selection	p::selection	Selects the portion of an element that is selected by a user

- **ATTRIBUTE SELECTOR**
  - The [attribute] selector is used to select elements with a specified attribute.
- **CSS [attribute="value"] Selector**
  - The [attribute="value"] selector is used to select elements with a specified attribute and value.
- **CSS [attribute~="value"] Selector**
  - The [attribute~="value"] selector is used to select elements with an attribute value containing a specified word.
- **CSS [attribute|="value"] Selector**
  - The [attribute|="value"] selector is used to select elements with the specified attribute, whose value can be exactly the specified value, or the specified value followed by a hyphen (-).
- **CSS [attribute^="value"] Selector**
  - The [attribute^="value"] selector is used to select elements with the specified attribute, whose value starts with the specified value.
- **CSS [attribute$="value"] Selector**
  - The [attribute$="value"] selector is used to select elements whose attribute value ends with a specified value.
- **CSS [attribute*="value"] Selector**
  - The [attribute*="value"] selector is used to select elements whose attribute value contains a specified value.

Selector	Example	Example description
[*attribute*]	[target]	Selects all elements with a target attribute
[*attribute=value*]	[target="_blank"]	Selects all elements with target="_blank"
[*attribute~=value*]	[title~="flower"]	Selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower"
[*attribute|=value*]	[lang|="en"]	Selects all elements with a lang attribute value starting with "en"
[*attribute^=value*]	a[href^="https"]	Selects all <a> elements with a href attribute value starting with "https"
[*attribute$=value*]	a[href$=".pdf"]	Selects all <a> elements with a href attribute value ending with ".pdf"
[*attribute*=value*]	a[href*="w3schools"]	Selects all <a> elements with a href attribute value containing the substring "w3schools"

!important

- The !important rule in CSS is used to add more importance to a property/value than normal.
- The only way to override an !important rule is to include another !important rule on a declaration with the same (or higher) specificity in the source code - and here the problem starts! This makes the CSS code confusing and the debugging will be hard, especially if you have a large style sheet!
- One way to use !important is if you have to override a style that cannot be overridden in any other way. This could be if you are working on a Content Management System (CMS) and cannot edit the CSS code. Then you can set some custom styles to override some of the CMS styles.

CSS Functions

- CSS functions are used to set the various CSS property. For example, the attr() function is used to retrieve the value of the HTML attribute.
- REFERENCE LINK https://www.geeksforgeeks.org/css-functions-complete-reference/
- **SOME OF THE CSS FUNCTIONS**
    - translateZ() - Reposition the element along the z-axis in 3D space.
    - var() - Insert a value for custom property.
    - url() - It is used to include a file.
    - translateY() - Reposition the element along the vertical axis.
    - translateX() - Reposition the element along the horizontal axis.
    - translate3d() - Reposition an element in 3D space.
    - translate() - Reposition an element in a horizontal and vertical direction.
    - skewY() - Transform an element in the vertical direction in a 2D plane.
    - skewX() - Transform an element in the horizontal direction in a 2D plane.
    - skew() - It is used to transform an element in the 2D plane.
    - sepia() - Apply a filter to the image to convert an image into a sepia image.
    - scaleZ() - Resize an element along the z-axis.
    - scaleY() - Resize an element along the y-axis in a 2D plane.
    - scaleX() - Resize an element along the x-axis in a 2D plane.
    - scale3d() - Resize the element in a 3D space. It scales the elements in x, y, and z planes.

- scale() - It is used to resize the element in a 2D plane.
- saturate() - It is used to super-saturate or desaturate the input image.
- rotateZ() - Rotate an element around the z-axis.
- rotateY() - Rotate an element around the vertical axis.
- rotateX() - Rotate an element around the x-axis.
- rotate3d() - Set the style in web pages that contain HTML elements.
- rotate() - Rotate an element based on the given angle as an argument.
- rgba() - It is used to define the colors using the Red-Green-Blue-Alpha (RGBA) model.
- rgb() - It is used to define the colors using the Red Green Blue (RGB) model.
- repeating-radial-gradient() - It is used to repeat radial gradients.
- repeating-linear-gradient() - It is used to repeat linear gradients.
- repeating-conic-gradient() - Repeat conic gradients in the background image.
- radial-gradient() - Set a radial gradient as the background image.
- polygon() - It is used with the filter property to create a polygon of images or text.
- perspective() - It is used with the transform property to set the perspective effect on images.
- opacity() - Apply a filter to the image to set the transparency of the image.
- min() - Return the minimum value from a set of comma-separated values.
- max() - Return the largest value from a set of comma-separated values.
- matrix3d() - Create a 3D transformation as a 4×4 homogeneous matrix.
- matrix() - Create a homogeneous 2D transformation matrix.
- linear-gradient() - Set the linear gradient as the background image.
- invert() - Set the invert of the color of the sample image.
- inset() - It is used with the filter property to change the inset of images.
- hue-rotate() - Apply a filter to the image to set the hue rotation of the image.
- hsla() - Define the colors using the Hue Saturation Lightness Alpha (HSLA) model.
- hsl() - Define the colors using the Hue-saturation-lightness model (HSL).
- grayscale() - Apply a filter to the image to set the grayscale of the image.
- env() - Insert the value of a user agent-defined environment variable into your CSS.
- ellipse() - Create floating text around the ellipse shape picture or anything else.
- drop-shadow() - Apply a filter to the image to set the shadow of the image.
- cubic-bezier() - Define a Cubic Bezier curve.
- contrast() - Apply a filter to set the contrast of the image.
- conic-gradient() - Set a conic gradient as the background image.
- circle() - Create floating text around the circular shape picture or anything else.
- calc() - Returns the value of an attribute of the selected elements.
- brightness() - Apply a filter to set the brightness of the image.

    - blur() - Apply a blurred effect filter on the image.
    - attr() - Returns the value of an attribute of the selected elements.

Comments

- HTML comments are used to insert notes or explanations within the HTML code. These comments are not visible to users on the webpage but can be seen in the HTML source code.
- Syntax: HTML comments start with <!-- and end with -->.
- Explaining sections of code for easier understanding and maintenance.Temporarily disabling a section of code during development or testing.
- Providing notes for other developers working on the same project.