Lecture 17

CSS

tecnsoltrainings.com

CSS 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. The descendant selector matches all elements that are descendants of a specified element.

```
div p {
  background-color: yellow;
}
```

Child Selector (>)

The child selector selects all elements that are the children of a specified element. The following example selects all <p> elements that are children of a <div> element:

```
div > p {
  background-color: yellow;
}
```

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". The following example selects the first <p> element that are placed immediately after <div> elements:

```
div + p {
 background-color: yellow;
}
```

General Sibling Selector (~)

The general sibling selector selects all elements that are next siblings of a specified element. The following example selects all <p> elements that are next siblings of <div> elements:

```
div ~ p {
 background-color: yellow;
}
```

tecnsoltrainings.com

CSS Pseudo-classes

A pseudo-class is used to define a special state of an element. For example, it can be used to:

- Style an element when a user mouses over it
- Style visited and unvisited links differently
- Style an element when it gets focus

The syntax of pseudo-classes:

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

Anchor Pseudo-classes

Links can be displayed in different ways:

```
a:link {
 color: #FF0000;
}
a:visited {
 color: #00FF00;
}
a:hover {
  color: #FF00FF;
}
a:active {
  color: #0000FF;
}
```

Pseudo-classes and HTML Classes

Pseudo-classes can be combined with HTML classes:
When you hover over the link in the example, it will change color:

```
a.highlight:hover {
  color: #ff0000;
}
```

An example of using the :hover pseudo-class on a <div> element:

```
div:hover {
 background-color: blue;
}
```

Simple Tooltip Hover

Hover over a <div> element to show a <p> element (like a tooltip):

```
p {
 display: none;
 background-color: yellow;
 padding: 20px;
}

div:hover p {
 display: block;
}
```

The :first-child pseudo-class matches a specified element that is the first child of another element. In the following example, the selector matches any <p> element that is the first child of any element:

```
p:first-child {
  color: blue;
}
```

In the following example, the selector matches the first <i> element in all <p> elements:

```
p i:first-child {
  color: blue;
}
```

tecnsoltrainings.com

CSS Pseudo-elements

A CSS pseudo-element is used to style specified parts of an element For example, it can be used to:

- **Style the first letter, or line, of an element**
- **Insert content before, or after, the content of an element**

The syntax of pseudo-elements:

```
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 following example formats the first line of the text in all <p> elements:

```
p::first-line {
  color: #ff0000;
  font-variant: small-caps;
}
```

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

tecnsoltrainings.com

Pseudo-elements and HTML Classes

Pseudo-elements can be combined with HTML classes:

```
 p.intro::first-letter {
  color: #ff0000;
 font-size: 200%;
}
```

Multiple Pseudo-elements

Several pseudo-elements can also be combined. In the following example, the first letter of a paragraph will be red, in an xx-large font size. The rest of the first line will be blue, and in small-caps. The rest of the paragraph will be the default font size and color:

```
p::first-letter {
  color: #ff0000;
  font-size: xx-large;
}
```

```
p::first-line {
  color: #0000ff;
  font-variant: small-caps;
}
```

CSS - The ::before Pseudo-element

The ::before pseudo-element can be used to insert some content before the content of an element. The following example inserts an image before the content of each <h1> element:

```
h1::before {
  content: url(smiley.gif);
}
```

CSS - The ::after Pseudo-element

The ::after pseudo-element can be used to insert some content after the content of an element. The following example inserts an image after the content of each <h1> element:

```
h1::after {
  content: url(smiley.gif);
}
```

CSS - The ::marker Pseudo-element

The ::marker pseudo-element selects the markers of list items. The following example styles the markers of list items:

```
::marker {
  color: red;
 font-size: 23px;
}
```

CSS - The ::selection Pseudo-element

The ::selection pseudo-element matches the portion of an element that is selected by a user. The following CSS properties can be applied to ::selection: color, background, cursor, and outline. The following example makes the selected text red on a yellow background:

```
::selection {
  color: red;
 background: yellow;
}
```

tecnsoltrainings.com

CSS Attribute Selectors

The [attribute] selector is used to select elements with a specified attribute. The following example selects all <a> elements with a target attribute:

```
a[target] {
 background-color: yellow;
}
```

tecnsoltrainings.com

CSS [attribute="value"] Selector

The [attribute="value"] selector is used to select elements with a specified attribute and value. The following example selects all <a> elements with a target="_blank" attribute:

```
a[target="_blank"] {
 background-color: yellow;
}
```

CSS [attribute~="value"] Selector

The [attribute~="value"] selector is used to select elements with an attribute value containing a specified word. The following example selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower":

```
[title~="flower"] {
 border: 5px solid yellow;
}
```

The example above will match elements with title="flower", title="summer flower", and title="flower new", but not title="my-flower" or title="flowers".

tecnsoltrainings.com

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 (-). The value has to be a whole word, either alone, like class="top", or followed by a hyphen(-), like class="top-text".

```
[class|="top"] {
 background: yellow;
}
```

CSS [attribute^="value"] Selector

The [attribute^="value"] selector is used to select elements with the specified attribute, whose value starts with the specified value. The value does not have to be a whole word! The following example selects all elements with a class attribute value that starts with "top":

```css
[class^="top"] {
  background: yellow;
}
```

CSS [attribute$="value"] Selector

The [attribute$="value"] selector is used to select elements whose attribute value ends with a specified value. The following example selects all elements with a class attribute value that ends with "test":

```
[class$="test"] {
  background: yellow;
}
```

tecnsoltrainings.com

CSS [attribute*="value"] Selector

The [attribute*="value"] selector is used to select elements whose attribute value contains a specified value. The following example selects all elements with a class attribute value that contains "te":

```
[class*="te"] {
 background: yellow;
}
```

Styling Forms

The attribute selectors can be useful for styling forms without class or ID:

```
input[type="text"] {
  width: 150px;
 display: block;
 margin-bottom: 10px;
 background-color: yellow;
}

input[type="button"] {
  width: 120px;
  margin-left: 35px;
  display: block;
}
```