**HTML5 TAGS**

<hr> defines a thematic break in an HTML page (e.g. a shift of topic). element is most often displayed as a horizontal rule that is used to separate content (or define a change) in an HTML page.

**How to Build a HTML5 form**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| <form> tag is used to create an HTML form for user input. element can contain one or more of the following form elements:   * <input> * <textarea> * <button> * <select> * <option> * <optgroup> * <fieldset> * <label> * <output>   **Attributes:**   |  |  |  | | --- | --- | --- | | **Attribute** | **Value** | **Description** | | accept-charset | character\_set | Specifies the character encodings that are to be used for the form submission | | action | *URL* | Specifies where to send the form-data when a form is submitted | | autocomplete | on off | Specifies whether a form should have autocomplete on or off | | enctype | application/x-www-form-urlencoded multipart/form-data text/plain | Specifies how the form-data should be encoded when submitting it to the server (only for method="post") | | method | get post | Specifies the HTTP method to use when sending form-data | | name | text | Specifies the name of a form | | novalidate | novalidate | Specifies that the form should not be validated when submitted | | rel | external help license next nofollow noopener noreferrer opener prev search | Specifies the relationship between a linked resource and the current document | | target | \_blank \_self \_parent \_top | Specifies where to display the response that is received after submitting the form | |

SELECTORS CSS3

.div {

box-sizing: border-box;

}

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CSS Attribute Selectors**   * It is possible to style HTML elements that have specific attributes or attribute values  |  |  |  | | --- | --- | --- | | **Selector** | **example** | **Example description** | | [attribute] | [target] | Selects all elements with a target attribute | | [attribute=value]t | [target=\_blank] | Selects all elements with target="\_blank" | | [attribute~=value] | [title~=flower] | Selects all elements with a title attribute containing the word "flower" | | [[attribute|=value] | [lang|=en] | Selects all elements with a lang attribute value starting with "en" | | [attribute^=value] | a[href^="https"] | Selects every <a> element whose href attribute value begins with "https" | | [attribute$=value] | a[href$=".pdf"] | Selects every <a> element whose href attribute value ends with ".pdf | | [attribute\*=value] | a[href\*="w3schools"] | Selects every <a> element whose href attribute value contains the substring "w3schools" | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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  |  |  |  | | --- | --- | --- | | **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 | :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 |   **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  |  |  |  | | --- | --- | --- | | **Selector** | **example** | **Example description** | | ::after | p::after | Insert content after every <p> element | | ::before | p::before | Insert content before every <p> element | | ::first-letter | p::first-letter | Selects the first letter of every <p> element | | ::first-line | p::first-line | Selects the first line of every <p> element | | ::selection | p::selection | Selects the portion of an element that is selected by a user | | look here for more |  | <https://developer.mozilla.org/pt-BR/docs/Web/CSS/Pseudo-elements> | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CSS Specify**  What is specify?  If there are two or more CSS rules that point to the same element, the selector with the highest specificity value will "win", and its style declaration will be applied to that HTML element. Think of specificity as a score/rank that determines which style declaration are ultimately applied to an element.   |  |  |  | | --- | --- | --- | | **Selector** | **Specificity Value** | **Calculation** | | p | 1 | 1 | | p.test | 11 | 1 + 10 | | p#demo | 101 | 1 + 100 | | <p style="color:pink;"> | 1000 | 1000 | | #demo | 100 | 100 | | .test | 10 | 10 | | p.test1.test2 | 21 | 1 + 10 + 10 | | #navbar p#demo | 201 | 100 + 1 + 100 | | \* | 0 | 0 (the universal selector is ignored) |   **Note: Inline style gets a specificity value of 1000, and is always given the highest priority!**  **Note 2: There is one exception to this rule: if you use the !important rule, it will even override inline styles!** |