CSS SELECTOR

<h2>CSS Simple Selectors</h2>

<ol type ='a'>

<li>Universal Selector (\*) - it matches elements of any type. Each set of tags represents an element on the page.</li>

Syntax: \*

<li>Type Selector - must match one or more HTML elements of the same name. Thus, a selector of nav would match all HTML nav elements, and a selector of ul would match all HTML unordered lists, or ul elements.</li>

Syntax: element

<li>ID Selectors - is declared using a hash, or pound symbol (#) preceding a string of characters. The string of characters is defined by the developer. This selector matches any HTML element that has an ID attribute with the same value as that of the selector, but minus the hash symbol.</li>

Syntax: #id\_value

<li>Class Selectors - is the most useful of all CSS selectors. It is declared with a dot preceding a string of one or more characters</li>

Syntax: .class\_name <li>

Attribute Selectors - targets elements based on the presence and/or value of HTML attributes, and is declared using square brackets:</li>

<ul> <li>[attr] - represents an element with an attribute name of attr.</li>

<li>[attr=value] - represents an element with an attribute name of attr whose value is exactly value.</li>

<li>[attr~=value] - represents an element with an attribute name of attr whose value is a whitespace-separated list of words, one which is exactly value.</li>

<li>[attr^=value] - represents an element with an attribute name of attr whose value is prefixed preceded by value.</li>

</ul> <li>Pseudo - Classes - uses a colon character to identify a pseudo-state that an element might be in.</li> Example: the state of being hovered, or the state of being activated </ol> <hr>

**Pattern matching**

In CSS, pattern matching rules determine what style rules can apply to a specific elements in the [document tree](https://www.w3.org/TR/CSS22/conform.html#doctree). Patterns are called selectors that may range to a simple element names to valuable contextual patterns. When conditions in the pattern are true to a certain element the selector will match the a element.

**Selector syntax**

Simple selector may consider as a [type selector](https://www.w3.org/TR/CSS22/selector.html#type-selectors) or [universal selector](https://www.w3.org/TR/CSS22/selector.html#universal-selector) that followed immediately by zero o more different [attribute selectors](https://www.w3.org/TR/CSS22/selector.html#attribute-selectors), [ID selectors](https://www.w3.org/TR/CSS22/selector.html#id-selectors), or [pseudo-classes](https://www.w3.org/TR/CSS22/selector.html#pseudo-classes), in any order. All simple selector matches when the components match to it.

A selector is a chain of one or more simple selectors separated by combinators. Combinators are the white space, ">", and "+". White space may be seen in between a combinator and the different simple selectors that surround it

**Universal selector**

Universal selector are always written in a "\*" that matches in the name of different element type. It matches any single element in the [document tree.](https://www.w3.org/TR/CSS22/conform.html#doctree)

**Type selectors**

Type selector matches the name of a document language element type. A type selector matches in every instance of the element type in the document tree.

**Descendant selectors**

Users may want selectors to correspond to a element which is descendant of another element in the document tree . Descendant selectors conveys relationship in a pattern. Descendant selector is made up of two or more selectors separated by [white space](https://www.w3.org/TR/CSS22/syndata.html#whitespace).

**Attribute selectors**

It allows users to specify rules that match to the elements which have certain attributes defined in the source document.

**ID selectors**

It may allow users to assign an identifier to a specific element instance in the document tree. CSS ID selectors match an element instance based on its identifier. A CSS ID selector always contains a "#" immediately followed by the ID value, which must be an identifier.

**Pseudo-elements and****pseudo-classes**

\*Pseudo-elements create abstractions about the document tree beyond those specified by the document language. For instance, document languages do not offer mechanisms to access the first letter or first line of an element's content.

\*Pseudo-classes classify elements on characteristics other than their name, attributes or content; in principle characteristics that cannot be deduced from the document tree. Pseudo-classes may be dynamic, in the sense that an element may acquire or lose a pseudo-class while a user interacts with the document.