

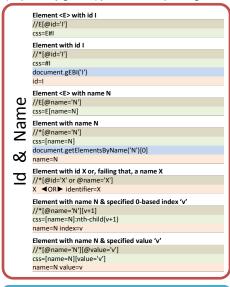


//E/@A ☒ {Se: //E@A } {Se: css=F@A } 0 document.gEBTN('E')[0].getAttribute('A') {Se: document.gEBTN('E')[0]@A } Attribute A of any element //*/@A 🖾 {Se://*@A } Attribute A1 of element <E> where attribute A2 is 't' exactly //E[@A2='t']/@A1 🗭 {Se: //E[@A2='t']@A1 } {Se: css=E[A2='t']@A1 } Attribute A of element <E> where A contains 't'

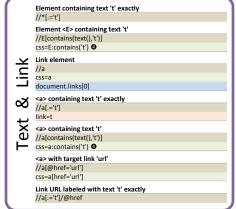
//E[contains(@A,'t')]/@A ☒ {Se: //E[contains(@A,'t')]@A }

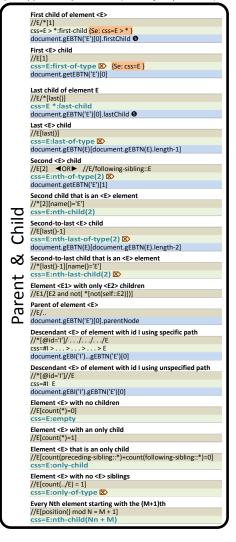
XPATH • GSS • DOM • SELENIUM Rosetta Stone and Cookbook

Sprinkled with Selenium usage tips, this is both a general-purpose set of recipes for each technology as well as a cross-reference to map from one to another. The validation suite for this reference chart (http://bit.ly/gTd5oc) provides example usage for each recipe supported by Selenium (the majority of them).



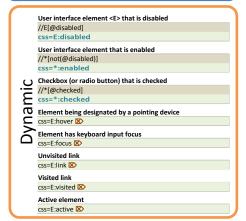
	Element <e> is explicitly in language L or subcode</e>
S	//E[@lang='L' or starts-with(@lang, concat('L', '-'))]
SE	css=E[lang =L]
Class	Element <e> is in language L or subcode (possibly inherited</e>
\circ	css=E:lang(L)
ď	Element with a class C
-0	//*[contains(concat('●', @class, '●'), '●C●')]
þΩ	css=.C
Пg	document.getElementsByClassName('C')[0]
a	Element <e> with a class C</e>
_	//E[contains(concat('⊙', @class, '⊙'), '⊙C⊙')]
	css=E.C





Element <E1> following some sibling <E2> //E2/following-sibling::E1 css=E2 ~ E1 Element <E1> immediately following sibling <E2> //E2/following-sibling::*[1][name()='E1'] Element <E1> following sibling <E2> with one intermediary //E2/following-sibling::*[2][name()='E1'] css=E2 + * + E1 Sibling element immediately following <E> plin //E/following-sibling::* css=F + 1 둜 document.gEBTN('E')[0].nextSibling 6 Element <E1> preceding some sibling <E2> //E2/preceding-sibling::E1 Element <E1> immediately preceding sibling <E2> //E2/preceding-sibling::*[1][name()='E1'] Element <E1> preceding sibling <E2> with one intermediary //E2/preceding-sibling::*[2][name()='E1'] Sibling element immediately preceding <E> //E/preceding-sibling::*[1] document.gEBTN('E2')[0].previousSibling 6

Cell by row and column (e.g. 3rd row, 2nd column) //*[@id='TestTable']//tr[3]//td[2] {Se: //*[@id='TestTable'].2.1 } css=#TestTable tr:nth-child(3) td:nth-child(2) {Se: css=#TestTable.2.1 } document.gEBI('TestTable').gEBTN('tr')[2].gEBTN('td')[1] {Se: document.gEBI('TestTable').2.1 } ᅙ Cell immediately following cell containing 't' exactly //td[preceding-sibling::td='t'] Cell immediately following cell containing 't' //td[preceding-sibling::td[contains(.,'t')]] css=td:contains('t') ~ td @



• DOM has limited capability with a simple 'document...' expression; however, arbitrary JavaScript code may be used as shown in this example CSS does not support qualifying elements with the style attribute, as in div[style*='border-width'].

Selenium uses a special syntax for returning attributes; normal XPath, CSS, and DOM syntax will fail.

4 CSS: The CSS2 contains function is not in CSS3; however, Selenium supports the superset of CSS1, 2, and 3.

5 DOM: firstChild, lastChild, nextSibling, and previousSibling are problematic with mixed content; they will point to empty text nodes rather than desired elements depending on whitespace in web page source.

{Se: ... } Selenium-only variation Not supported by Selenium Space character

expression CSS3 or XPath 2.0

DOM abbreviations:

gEBTN getElementsByTagName

gEBI getElementBvId

General

Indexing (all): XPath and CSS use 1-based indexing; DOM and Selenium's table syntax use 0-based indexing. Prefixes (all): |xpath=| required unless expression starts with // • |dom=| required unless expression starts with "document." • |css=| always required • |identifier=| never | required.

Cardinality (Selenium): XPath and CSS may specify a node set or a single node; DOM must specify a single node. When a node set is specified, Selenium returns just the first node. Content (XPath): Generally should use normalize-space() when operating on display text.



XPath CSS DOM Selenium