**Selectors**

**Shortkeys:**

**Link text 🡪 =**

**Partial link text 🡪 \*=**

**Class🡪 .**

**Id🡪#**

**Tag Name🡪< tagName />**

**Name🡪 name= ” “**

**Link text**

For example:

<a href="https://webdriver.io">WebdriverIO</a>

You can query this element by calling:

const link = await $('=WebdriverIO')

console.log(await link.getText()) // outputs: "WebdriverIO"

console.log(await link.getAttribute('href')) // outputs: "https://webdriver.io"

**Partial link text**

For example:

<a href="https://webdriver.io">WebdriverIO</a>

You can query this element by calling:

const link = await $('\*=driver')

console.log(await link.getText()) // outputs: "WebdriverIO"

**Element with certain text**

For example, here's a query for a level 1 heading with the text "Welcome to my Page":

<h1 alt="welcome-to-my-page">Welcome to my Page</h1>

You can query this element by calling:

const header = await $('h1=Welcome to my Page')  
console.log(await header.getText()) *// outputs: "Welcome to my Page"*  
console.log(await header.getTagName()) *// outputs: "h1"*

Or using query partial text:

const header = await $('h1\*=Welcome')  
console.log(await header.getText()) *// outputs: "Welcome to my Page"*

The same works for id and class names:

<i class="someElem" id="elem">WebdriverIO is the best</i>

You can query this element by calling:

const classNameAndText = await $('.someElem=WebdriverIO is the best')  
console.log(await classNameAndText.getText()) *// outputs: "WebdriverIO is the best"*  
  
const idAndText = await $('#elem=WebdriverIO is the best')  
console.log(await idAndText.getText()) *// outputs: "WebdriverIO is the best"*  
  
const classNameAndPartialText = await $('.someElem\*=WebdriverIO')  
console.log(await classNameAndPartialText.getText()) *// outputs: "WebdriverIO is the best"*  
  
const idAndPartialText = await $('#elem\*=WebdriverIO')  
console.log(await idAndPartialText.getText()) *// outputs: "WebdriverIO is*

**Note:** You can't mix multiple selector strategies in one selector. Use multiple chained element queries to reach the same goal, e.g.:

const elem = await $('header h1\*=Welcome') *// doesn't work!!!*  
*// use instead*  
const elem = await $('header').$('h1\*=Welcome')

Tag Name[​](https://webdriver.io/docs/selectors/#tag-name)

To query an element with a specific tag name, use <tag> or <tag />.

<my-element>WebdriverIO is the best</my-element>

You can query this element by calling:

const classNameAndText = await $('<my-element />')  
console.log(await classNameAndText.getText()) *// outputs: "WebdriverIO is*

## Name Attribute

<input name="username" value="foobar" />

Copy

const classNameAndText = await $('[name="username"]')  
console.log(await classNameAndText.getValue()) *// outputs: "foobar"*

ARIA - Role Attribute[​](https://webdriver.io/docs/selectors/#aria---role-attribute)

For querying elements based on [ARIA roles](https://www.w3.org/TR/html-aria/#docconformance), you can directly specify role of the element like [role=button] as selector parameter:

<button>Click me</button>

const button = await $('[role=button]')  
  
*// perform click on button element*  
console.log(await button.click())

Applications:

forEach loop:

it('should fetch menu links and visit each page', async () => {  
 const links = await $$('#menu a')  
 await links.forEach(async (link) => {  
 await link.click()  
 })  
})

map:

let pricelist=['$12','$15','$45','$74','$38','$86']

let pricesort=pricelist.map(ele=>ele.slice(1,3))

console.log(pricesort.sort((x,y)=>x-y));

console.log(pricelist);

[ '12', '15', '38', '45', '74', '86' ]

[ '$12', '$15', '$45', '$74', '$38', '$86' ]