**Methods in Cypress**

**get()**

## Syntax

cy.get(selector)

cy.get(alias)

cy.get(selector, options)

cy.get(alias, options)

### Usage

**Correct Usage**

cy.get('.list > li') // Yield the <li>'s in .list

### Arguments

**selector** **(String selector)**

A selector used to filter matching DOM elements.

**alias** **(String)**

An alias as defined using the [.as()](https://docs.cypress.io/api/commands/as) command and referenced with the @ character and the name of the alias.

You can use cy.get() for aliases of primitives, regular objects, or even DOM elements.

When using aliases with DOM elements, Cypress will query the DOM again if the previously aliased DOM element has gone stale.

**options** **(Object)**

Pass in an options object to change the default behavior of cy.get().

| **Option** | **Default** | **Description** |
| --- | --- | --- |
| log | true | Displays the command in the [Command log](https://docs.cypress.io/guides/core-concepts/test-runner#Command-Log) |
| timeout | [defaultCommandTimeout](https://docs.cypress.io/guides/references/configuration#Timeouts) | Time to wait for cy.get() to resolve before [timing out](https://docs.cypress.io/api/commands/get#Timeouts) |
| withinSubject | null | Element to search for children in. If null, search begins from root-level DOM element |
| includeShadowDom | [includeShadowDom config option value](https://docs.cypress.io/guides/references/configuration#Global) | Whether to traverse shadow DOM boundaries and include elements within the shadow DOM in the yielded results. |

### Yields

* cy.get() yields the DOM element(s) it found.

## Examples

### Selector

#### **Get the input element**

cy.get('input').should('be.disabled')

#### **Find the first li descendent within a ul**

cy.get('ul li:first').should('have.class', 'active')

#### **Find the dropdown-menu and click it**

cy.get('.dropdown-menu').click()

#### **Find 5 elements with the given data attribute**

cy.get('[data-test-id="test-example"]').should('have.length', 5)

#### **Find the link with an href attribute containing the word "questions" and click it**

cy.get('a[href\*="questions"]').click()

#### **Find the element with id that starts with "local-"**

cy.get('[id^=local-]')

#### **Find the element with id that ends with "-remote"**

cy.get('[id$=-remote]')

#### **Find the element with id that starts with "local-" and ends with "-remote"**

cy.get('[id^=local-][id$=-remote]')

#### **Find the element with id that has characters used in CSS like ".", ":".**

cy.get('#id\\.\\.\\.1234') // escape the character with \\

### Get in **.within()**

#### **cy.get() in the**[**.within()**](https://docs.cypress.io/api/commands/within)**command**

Since cy.get() is chained off of cy, it always looks for the selector within the entire document. The only exception is when used inside a [.within()](https://docs.cypress.io/api/commands/within) command.

cy.get('form').within(() => {

cy.get('input').type('Pamela') // Only yield inputs within form

cy.get('textarea').type('is a developer') // Only yield textareas within form

})

### Get vs Find

The cy.get command always starts its search from the [cy.root](https://on.cypress.io/root" \t "_blank) element. In most cases, it is the document element, unless used inside the [.within()](https://docs.cypress.io/api/commands/within) command. The [.find](https://on.cypress.io/find) command starts its search from the current subject.

<div class="test-title">cy.get vs .find</div>

<section id="comparison">

<div class="feature">Both are querying commands</div>

</section>

cy.get('#comparison')

.get('div')

// finds the div.test-title outside the #comparison

// and the div.feature inside

.should('have.class', 'test-title')

.and('have.class', 'feature')

cy.get('#comparison')

.find('div')

// the search is limited to the tree at #comparison element

// so it finds div.feature only

.should('have.length', 1)

.and('have.class', 'feature')

### Alias

## Rules

### Requirements

* cy.get() requires being chained off a command that yields DOM element(s).

### Assertions

* cy.get() will automatically [retry](https://docs.cypress.io/guides/core-concepts/retry-ability) until the element(s) [exist in the DOM](https://docs.cypress.io/guides/core-concepts/introduction-to-cypress#Default-Assertions)
* cy.get() will automatically [retry](https://docs.cypress.io/guides/core-concepts/retry-ability) until all chained assertions have passed

### Timeouts

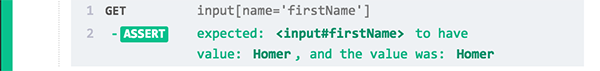
* cy.get() can time out waiting for the element(s) to [exist in the DOM](https://docs.cypress.io/guides/core-concepts/introduction-to-cypress#Default-Assertions) .
* cy.get() can time out waiting for assertions you've added to pass.

## Command Log

**Get an input and assert on the value**

cy.get('input[name="firstName"]').should('have.value', 'Homer')

The commands above will display in the Command Log as:



When clicking on the get command within the command log, the console outputs the following:



# **find()**

Get the descendent DOM elements of a specific selector.

The querying behavior of this command matches exactly how [.find()](http://api.jquery.com/find) works in jQuery.

## Syntax

.find(selector)

.find(selector, options)

### Usage

**Correct Usage**

cy.get('.article').find('footer') // Yield 'footer' within '.article'

**Incorrect Usage**

cy.find('.progress') // Errors, cannot be chained off 'cy'

cy.exec('node start').find() // Errors, 'exec' does not yield DOM element

### Arguments

**selector** **(String selector)**

A selector used to filter matching descendent DOM elements.

**options** **(Object)**

Pass in an options object to change the default behavior of .find().

| **Option** | **Default** | **Description** |
| --- | --- | --- |
| log | true | Displays the command in the [Command log](https://docs.cypress.io/guides/core-concepts/test-runner#Command-Log) |
| timeout | [defaultCommandTimeout](https://docs.cypress.io/guides/references/configuration#Timeouts) | Time to wait for .find() to resolve before [timing out](https://docs.cypress.io/api/commands/find#Timeouts) |
| includeShadowDom | [includeShadowDom config option value](https://docs.cypress.io/guides/references/configuration#Global) | Whether to traverse shadow DOM boundaries and include elements within the shadow DOM in the yielded results. |

### Yields

* .find() yields the new DOM element(s) it found.

## Examples

### Selector

#### **Get li's within parent**

<ul id="parent">

<li class="first"></li>

<li class="second"></li>

</ul>

// yields [<li class="first"></li>, <li class="second"></li>]

cy.get('#parent').find('li')

## Rules

### Requirements

* .find() requires being chained off a command that yields DOM element(s).

### Assertions

* .find() will automatically [retry](https://docs.cypress.io/guides/core-concepts/retry-ability) until the element(s) [exist in the DOM](https://docs.cypress.io/guides/core-concepts/introduction-to-cypress#Default-Assertions)
* .find() will automatically [retry](https://docs.cypress.io/guides/core-concepts/retry-ability) until all chained assertions have passed

### Timeouts

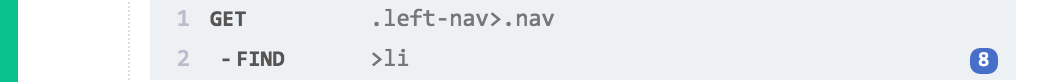
* .find() can time out waiting for the element(s) to [exist in the DOM](https://docs.cypress.io/guides/core-concepts/introduction-to-cypress#Default-Assertions) .
* .find() can time out waiting for assertions you've added to pass.

## Command Log console.log find

**Find the li's within the nav**

cy.get('.left-nav>.nav').find('>li')

The commands above will display in the Command Log as:



When clicking on the find command within the command log, the console outputs the following:

# **within()**

Scopes all subsequent cy commands to within this element. Useful when working within a particular group of elements such as a <form>.

## Syntax

.within(callbackFn)

.within(options, callbackFn)

### Usage

**Correct Usage**

cy.get('.list').within(($list) => {}) // Yield the `.list` and scope all commands within it

**Incorrect Usage**

cy.within(() => {}) // Errors, cannot be chained off 'cy'

cy.getCookies().within(() => {}) // Errors, 'getCookies' does not yield DOM element

### Arguments

**callbackFn** **(Function)**

Pass a function that takes the current yielded subject as its first argument.

**options** **(Object)**

Pass in an options object to change the default behavior of .within().

| **Option** | **Default** | **Description** |
| --- | --- | --- |
| log | true | Displays the command in the [Command log](https://docs.cypress.io/guides/core-concepts/test-runner#Command-Log) |

### Yields

* .within() yields the same subject it was given from the previous command.

Trying to return a different element the .within callback have no effect:

<div id="within-yields">

The parent div

<div class="some-child">Child element</div>

</div>

cy.get('#within-yields')

.within(() => {

// we are trying to return something

// from the .within callback,

// but it won't have any effect

return cy.contains('Child element').should('have.class', 'some-child')

})

.should('have.id', 'within-yields')

Similarly, trying to change the subject by using the [cy.wrap](https://docs.cypress.io/api/commands/wrap) command inside the .within callback have no effect:

<div id="wrap-inside-within">

The parent div

<div class="some-child">Child element</div>

</div>

cy.get('#wrap-inside-within')

.within(() => {

// returning cy.wrap(...) has no effect on the yielded value

// it will still be the original parent DOM element

return cy.wrap('a new value')

})

.should('have.id', 'wrap-inside-within')

## Examples

### Forms

#### **Get inputs within a form and submit the form**

<form>

<input name="email" type="email" />

<input name="password" type="password" />

<button type="submit">Login</button>

</form>

cy.get('form').within(($form) => {

// you have access to the found form via

// the jQuery object $form if you need it

// cy.get() will only search for elements within form,

// not within the entire document

cy.get('input[name="email"]').type('john.doe@email.com')

cy.get('input[name="password"]').type('password')

cy.root().submit()

})

### Tables

#### **Find row with specific cell and confirm other cells in the row**

<table>

<tr>

<td>My first client</td>

<td>My first project</td>

<td>0</td>

<td>Active</td>

<td><button>Edit</button></td>

</tr>

</table>

cy.contains('My first client')

.parent('tr')

.within(() => {

// all searches are automatically rooted to the found tr element

cy.get('td').eq(1).contains('My first project')

cy.get('td').eq(2).contains('0')

cy.get('td').eq(3).contains('Active')

cy.get('td').eq(4).contains('button', 'Edit').click()

})

### Temporarily escape

You can temporarily escape the .within context by starting a new command chain with [cy.root](https://docs.cypress.io/api/commands/root) followed by [.closest](https://docs.cypress.io/api/commands/closest) commands.

<section class="example">

<!-- note the input field outside the form -->

<input id="name" type="text" />

<form>

<input name="email" type="email" />

<input name="password" type="password" />

<button type="submit">Login</button>

</form>

</section>

cy.get('form').within(($form) => {

// temporarily escape the .within context

cy.root().closest('.example').find('#name').type('Joe')

// continue using the .within context

cy.get('input[name="email"]').type('john.doe@email.com')

cy.get('input[name="password"]').type('password')

cy.root().submit()

})

## Rules

### Requirements

* .within() requires being chained off a previous command.

### Assertions

* .within() will only run assertions you have chained once, and will not [retry](https://docs.cypress.io/guides/core-concepts/retry-ability) .

### Timeouts

* .within() cannot time out.

## Command Log

**Get the input within the form**

cy.get('.query-form').within((el) => {

cy.get('input:first')

})

The commands above will display in the Command Log as:



When clicking on the within command within the command log, the console outputs the following:

