Cypress Cheat Sheet—Part 2

Author: Anshita Bhasin
LinkedIn: https://www.linkedin.com/in/anshita-bhasin/

Below are some of the important commands to traverse the DOM elements in Cypress.

(1) .eq()

This command is used to select a (n+1)th element within a set of elements that match a given selector. It takes an index argument, which specifies the index of the element that should be selected. The index starts with 0.

Note that the index is zero-based, so eq(0) would retrieve the first element, eq(1) would retrieve the second element, and so on.

Below is a sample code:

```
cy.get('li').eq(1).should('have.text', 'Item 2')
```

In this example,we are retrieving the second li element and then verifying that its text content is 'Item 2'.

Here are some more examples of how to use .eq()

```
1) cy.get(.figure-img-wrapper).eq(1)
2) cy.get(.figure-img-wrapper).eq(1,{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).eq(1).type('testing')
4) cy.get(.figure-img-wrapper).eq(1).should('have.text','testing')
```

It is often used in conjunction with other commands, such as click() or type(), to perform actions on the specific element in a set.

(2) .first()

This command in Cypress is used to select the first element in a set of elements. It is similar to using the .eq(0) command, which also selects the first element in a set of elements.

Below is a sample code:

```
cy.get('li').first().should('have.text', 'Item 1')
```

In the above example, we are retrieving the first li element and checking that its text content is ‘Item 1’.

Here are some more examples of how to use .first()

```
1) cy.get(.figure-img-wrapper).first()
2) cy.get(.figure-img-wrapper).first({timeout:2000}).click()
3) cy.get(.figure-img-wrapper).first().type('testing')
4) cy.get(.figure-img-wrapper).first().should('have.text','testing')
```

It is often used in conjunction with other commands, such as click(),type(), or should, to perform actions on the first element in a set.

(3) .last()

This command is used to select the last element within a group of elements that match a specific selector. It can be useful for selecting and interacting with the last element within a group of elements.

Below is a sample code:

```
cy.get('li').last().should('have.text', 'Item 3')
```

In the above example, we are retrieving the last li element and checking that its text content is ‘Item 3’.

Here are some more examples of how to use .last()

```
1) cy.get(.figure-img-wrapper).last()
2) cy.get(.figure-img-wrapper).last({timeout:2000}).click()
3) cy.get(.figure-img-wrapper).last().type('testing')
4) cy.get(.figure-img-wrapper).last().should('have.text','testing')
```

It is often used in conjunction with other commands, such as click(), type(), or should, to perform actions on the first element in a set.

(4) .next()

This command in Cypress is used to get the next sibling element of the current DOM element.

Let’s understand with the below example:

```
cy.get('span').first().next().should('have.text', 'Item 2')
```

In the above example, we are retrieving the next sibling element of the first span element and checking that its text content is ‘Item 2’.

Here are some more examples of how to use .next()

```
1) cy.get(.figure-img-wrapper).next()
2)cy.get('.traversal-ul').contains('apples').next({timeout:2000}).should('c
ontain', 'oranges')
3) cy.get('.traversal-ul').next().should('contain', 'oranges')
4) cy.get('.traversal-ul').find(.active).next().should('contain', 'oranges')
5) cy.get('label.text-label').next().type('testing')
6) cy.get('label.text-label').next().click()
```

It is often used in conjunction with other commands, such as click(),type()or .should()

(5) .nextALL()

This command is used to get all the next siblings of an element. It is similar to the .next() method, which gets the next sibling element, but .nextAll() gets all the next siblings.

Let’s understand with below example:

```
cy.get('span').first().nextAll().should('have.length', 3)
```

In the above example, we are retrieving all of the next sibling elements of the first span element and checking that there are 3 of them.

Here are some more example of to use .nextAll():

```
1) cy.get(.figure-img-wrapper).nextAll().should('have.length',3)
2)cy.get('.traversal-ul').contains('apples').nextAll({timeout:2000}).should
('have.length',3)
```

(6) .nextUntil()

This command is used to retrieve the next sibling elements of the selected element until the specified selector is met. This command is particularly useful when you have a set of elements that are not necessarily adjacent, but you want to perform the same action on all of them.

Let’s understand with below example:

```
cy.get('span').first().nextUntil('.stop').should('have.length', 2)
```

In the above example, we are retrieving the next element sibling elements of the first span element until the element with the class stop is reached and checking that there are 2 of them.

Here are some more examples of how to use .nextUntil():

```
1) cy.get(.figure-img-wrapper).nextUntill('.data')
2)cy.get(.figure-img-wrapper).nextUntill('.data',{timeout:2000}).should('
have.length',3)
```

(7) .parent()

This command is used to retrieve the parent element of the currently selected element. This is useful when you need to perform an action on the parent element of an element, rather than the element itself.

Let’s understand with below example:

```
cy.get('.child-element').parent().should('have.class', 'parent-element')
```

In the above example, we are trying to select an element with the class 'child-element', then it is navigating to the parent element and asserting that the parent element has the class 'parent-element'.

Here are some more examples of how to use .parent():

```
1) cy.get(.figure-img-wrapper).parent({timeout:2000})
2) cy.get(.figure-img-wrapper).parent('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).parent().type('testing')
4) cy.get(.figure-img-wrapper).parent().should('have.length',2)
```

Keep in mind that the .parent() command only works on the currently selected element, so you’ll need to make sure that you’ve selected the correct element before using it.

(8) .parents()

This command is used to traverse through the ancestor elements of a selected DOM element. It is similar to the .parent() command, but instead of returning just the immediate parent element, it returns all of the parent elements from the immediate parent up to the root element of the DOM.

Let’s understand with below example:

```
cy.get('.child-element').parents().should('have.length', 2)
```

In the above example, we are selecting an element with the class 'child-element', then it is navigating to all the parent elements of the selected element, including grandparents and asserting that the number of elements in the collection of all parent elements is 2.

Here are some more examples of how to use .parents():

```
1) cy.get(.figure-img-wrapper).parents({timeout:2000})
2) cy.get(.figure-img-wrapper).parents('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).parents().type('testing')
4) cy.get(.figure-img-wrapper).parents().should('have.length',2)
```

(9) .parentsUntil():

This command is used to retrieve all the parent elements of a given DOM element up to, but not including, the element specified in the argument.

This command can be useful when you want to traverse up the DOM tree to find a specific ancestor element.

Let’s understand with below example:

```
cy.get('.child-element').parentsUntil('.grandparent-element').should('have.length', 1)
```

In the above example, we are trying to select the first element with class child-element and then using the parentsUntil method to select all the parent elements of child-element until the grandparent-element is encountered, then asserting that the selection should have exactly one element. This command is checking whether there is one parent element of the child-element element, that is not the grandparent-element.

Here are some more examples of how to use parentsUntil() :

```
1) cy.get(.figure-img-wrapper).parentsUntil({timeout:2000})
2) cy.get(.figure-img-wrapper).parentsUntil('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').parentsUntil('.active')
4) cy.get(.figure-img-wrapper).parentsUntil().should('have.length',2)
```

(10) .children()

This command is used to select the children of the DOM element(s) that are currently selected. It returns a new Cypress object containing these child elements.

Let’s understand with below example:

```
cy.get('.parent-element').children().should('have.length', 2)
```

In the above example we are trying to find the first element with class parent-element, then it's getting all it's child elements and finally asserting that this selection should have exactly two children elements. This command is essentially checking whether there are 2 children of the parent-element element.

Here are some more examples of how to use .children():

```
1) cy.get(.figure-img-wrapper).children({timeout:2000})
2) cy.get(.figure-img-wrapper).children('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').children('.active')
4) cy.get(.figure-img-wrapper).children().should('have.length',2)
```

(11) .siblings()

This command is used to select a sibling element that is a direct child of the same parent element as the current element. It takes a selector argument to specify which sibling element to select.

Let's understand with below example:

```
cy.get('.child-element-2').siblings().should('have.length', 2)
```

In the above example, we are trying to select the first element with class child-element-2 and then using the siblings method to select all the siblings elements of the selected element and then it's asserting that this selection should have 2 elements. This command is essentially checking whether there are 2 siblings elements of the child-element-2 element.

Here are some more example of how to use .sibling() command:

```
1) cy.get(.figure-img-wrapper).siblings({timeout:2000})
2) cy.get(.figure-img-wrapper).siblings('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').siblings('.active')
4) cy.get(.figure-img-wrapper).siblings().should('have.length',2)
```

(12) .prev()

This command is used to get the immediately preceding sibling of an element. It only gets the first preceding sibling.

Let's understand with below example:

```
cy.get('.child-element-3').prev().should('have.class', 'child-element-2')
```

In the above example, we are trying to select the first element with class child-element-3, and using the prev method to select it's previous sibling element and then it's asserting that this previous sibling element should have a class child-element-2. This command is essentially checking whether the element immediately before child-element-3 has the class child-element-2.

Here are some more examples of how to use .prev() in a test:

```
1) cy.get(.figure-img-wrapper).prev({timeout:2000})
2) cy.get(.figure-img-wrapper).prev('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').prev('.active')
4) cy.get(.figure-img-wrapper).prev().should('have.length',2)
5) cy.get('.birds').find('.active').prev().should('contain','oranges')
```

(13) .prevAll()

This command is used to select all sibling elements that come before the selected element in the DOM tree.

Below is the sample code:

```
cy.get('.item.active').prevAll('.item')
```

In the above example, we are selecting the element with class item active and then using the prevAll method to select all the previous siblings elements that have the class item of the selected active item.

Here are few more examples of how to use .prevAll() in a test:

```
1) cy.get(.figure-img-wrapper).prevAll({timeout:2000})
2) cy.get(.figure-img-wrapper).prevAll('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').prevAll('.active')
4) cy.get(.figure-img-wrapper).prevAll().should('have.length',2)
5) cy.get('.birds').find('.active').prevAll().should('contain','oranges')
```

(14) .prevUntil()

This command is used to find all the preceding siblings of an element that match certain criteria, starting from the element that immediately precedes it and going backward. It is useful for finding elements that are not immediately adjacent to the element you are targeting.

Below is a sample code:

```
cy.get('.child-element-3').prevUntil('.child-element-1').should('have.length', 1)
```

In the above example, we are selecting the first element with class child-element-3 and then using the prevUntil method to select all previous siblings of child-element-3 until the child-element-1 is encountered, then asserting that the selection should have exactly one element.

Here are some more sample examples of how to use .prevUntil()

```
1) cy.get(.figure-img-wrapper).prevUntil({timeout:2000})
2) cy.get(.figure-img-wrapper).prevUntil('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).find('.active').preUntil('.active')
4) cy.get(.figure-img-wrapper).prevUntil().should('have.length',2)
5) cy.get('.birds').find('.active').prevUntil().should('contain','oranges')
```

(15) .filter()

It is used to select only those elements that match the provided selector.

Let's understand with below example:

```
cy.get('.parent-element').children().filter('.child-element-2').should('have.
length', 1)
```

In the above example,we are trying to find the first element with class parent-element, then it's getting all it's child elements, and then filtering all child elements with class child-element-2 and finally asserting that this selection should have exactly one element.

Here are some more examples of how to use .filter()

```
1) cy.get(.figure-img-wrapper).filter({timeout:2000})
2) cy.get(.figure-img-wrapper).filter('.active',{timeout:2000}).click()
3) cy.get(.figure-img-wrapper).filter('.active').type('apple')
4) cy.get(.figure-img-wrapper).filter().should('have.length',2)
5) cy.get('.birds').find('>li').filter('.active').should('contain', 'oranges');
```

Conclusion:

Above shared are the most used Cypress commands which are used to traverse the DOM elements. All the above commands cannot be used with commands like cy.getCookies(), cy.clock(), and cy.refresh()

Ref : https://docs.cypress.io/api/table-of-contents

Thanks for reading. Happy Learning!

~Anshita Bhasin