Cypress commands: Cheat Sheet

cy.visit

similary to driver.get() or navigate() in Selenium Webdriver. To launch a url in the browser.

cy.get('csslocator')

get is like findElement, identifies the element in the web page based on css provided.

cy.get('csslocator', {timeout: <timeinmilliseconds>})

when finding an element, adding an additional parameter "timeout" makes cypress to wait for the specified time interval until the element is visible.

cy.wait(time)

similar to thread.sleep

cy.type('learner')

enters text in the textbox

cy.get('#abc:visible')

retrieves only visible elements

cy.get('#abc').find('#a')

finds the elements from the parent locator used in get()

get<->find → acts as a parent/child chaining.

cy.get('#abc').find('#a').eq(2).contains('add to cart').click()

identifies the element equal with index 2 and clicks it

cy.get('#abc').find('#a').each(($el,index,$list)=>{

const text=$el.find('.h4').text()

if((text.contains('learner')){

$el.find('button').click()

}

}

iterates through for each of the elements located. get text for the element and if text matches the expected, clicks on the element.

cy.log('learner');

prints to cypress runner

Invoke Event with jquery code:

```
cy.get(‘csslocator’).then(function($input){
$input[0].setAttribute(‘value’, ‘cypress’)
}) .should(‘have.attr’, ‘value’, ‘cypress’)
```

Invoke Event with javascript code:

```
cy.get(‘csslocator’)
.invoke(‘attr’, ‘value’, ‘cypress’)
.should(‘have.attr’, ‘value’, ‘cypress’
```

Cypress asynchronous nature:

— ->

all commands hit the server at the same time.no guarantee in the sequence of execution.

then()

It is a promise.it waits until the step is completed.

```
//the below way wont work as cypress adds promise to each statement
const logoObject=cy.get(‘.branch’)
console.log(logoObject.text())

//the below way works
cy.get(‘.brand’).then(function(logoElement){
cy.log(logoElement.text())

)}
```

text() method is from Jquery, text() method is not part of cypress

Handling Async with promises:

— ->

we use alias ‘**as**’ instead of rewriting the same css for a element.

```
cy.get(‘#abc’).as(‘abcObject’)

cy.get(‘@abcObject’).find(‘.product’).type(‘bread’);
```

Difference b/w console.log and cy.log:

— ->

console.log- javascript related, asynchronous

cy.log- cyrpess method is handled in synchronous way

Should():

should is from chai library

cy.get('.branch').should('have.text','GROCERYSTORE')

contains():

cy.contains('CHECKOUT').click()

→ contains identifies the element with the specified text.

how to suppress cross origin domain issue?

— ->

add the following configuration in cypress.json file

settings{

chromeWebSecurity: false

}

Handle checkboxes:

— ->

check():

cy.get('#abc').check().should('be.checked').and('have.value','option1′)

for behavior things we should be use 'be'

uncheck():

cy.get('#abc').uncheck().should('not.be.checked')

check multiple checkboxes:

— ->

cy.get('input[type="checkbox"]).check(['option2','option1′])

Handling dropdowns:

— ->

static dropdowns:

— ->

select supports value or text as parameter cy.get(‘select’).select(‘option1’).should(‘have.value’,’option1′)

dynamic dropdowns:

— —->

enter the value and click on the value.

cy.get(‘#country’).type(‘dev’).each(($el,index,$list)=>{

if($el.text()===”Dev”){

$el.click()

}

)}

$el.get(‘#country’).should(‘have.value’,’Dev’)

Handling visible and invisible elements:

— —->

cy.get(‘#displayed-text’).should(‘be.visible’)

cy.get(‘#hide-textbox’).click()

cy.get(‘#displayed-text’).should(‘not.be.visible’)

cy.get(‘#show-textbox’).click()

cy.get(‘#displayed-text’).should(‘be.visible’)

Radio buttons:

— —->

cy.get(‘[value=”radio”]’).check().should(‘be.checked’)

Alerts:

— —->

cypress auto accepts alerts and popups(clicks on ok/accept button) cy.get(‘#alert’).click() //which opens only ok button alert

cy.get(‘[value=”Confirm”]’).click()// which opens ok/cancel button alert but cypress clicks on confirm button

cypress has the capability to listen to browser events and fires the browser events
– takes control of dom and manipulates the dom of the browser

–**window:alert** is the event which gets fired on alert open

cy.on(‘window:alert’,(str)=>{

//mocha

expect(str).to.equal(‘Hello there’);

})

–**window:confirm** is the event which gets fired on confirm popup

cy.on(‘window:confirm’,(str)=>{

//mocha

expect(str).to.equal(‘Bye’);

})

Handling child tab:

— ->

we can’t switch child taps using cypress directly.but we have a work around to overcome this issue.
we need to manipulate the dom to open the link in the same window and remove the target attribute

invoke (): invokes jquery function

— ->

cypress supports all jquery functions.

cy.get(‘#opentab’).invoke(‘removeAttr’,’target’).click();

navigating browser controls:

— ->

cy.go(‘back’);//navigate to the previous page cy.url().should(‘include’,’abc.com’)

cy.go(‘forward’);//navigate forward

handling web tables:

— ->

cy.get(‘tr td:nth-child(2)’).each(($el,index,$list)=>{

const rowText= $el.text()

if(rowText.includes(‘mango’))

{

cy.get(“tr td:nth-child(2)”).eq(index).next()

.then(function(price){

const priceText= price.text()

expect(priceText).to.equal(‘23’)

}

}

}

traverse to sibling elements:

— ->

next() gets the immediate following sibling of the element. This works only on top of get method.

cy.get(‘#link1’).next()

prev() gets the immediate preceding sibling of the element

cy.get(‘#t1’).prev()

similarly we have prevAll(),nextAll(),prevUntil(),nextUntil()

siblings() gets the siblings of dom elements.

cy.get(‘#id1’).siblings(‘.test’)

Handling mouse hover:

— ->

mouse hover events are not supported by default, work around is to use jquery or force click.

cy.get('div.mouse-hover-content').invoke('show')

cy.contains('Top').click()

(or)

cy.contains('Top').click({force:true})

cy.url().should('include','top')

force a click regardless of it's action state-

cy.get('button').click({force:true})

Handling child windows:

— ->

Just like child tabs, child windows handling can't be done

using cypress.

child window/tab:

1st method-

remove target attribute of the a tag

2nd method-

href attribute

prop(attr) is a jquery method

cy.get('#opentab').then(function(el){

const url=el.prop('href')

cy.log(url)

cy.visit(url)//fails if the domain of website is different

//to handle this issue

}

Closing browser windows:

— ->

cypress automatically closes the browser once the test is being executed. we need not explicitly close the browser window.

————————————————————————————->