Cần tìm hiểu thêm:

+ its()

+ wrap()

+ and()

+ describe()

+ Tìm hiểu ý nghĩa của các chai như: include, …

E2E Testing:

+ Test web navigation (đường dẫn web)

+ Test DOM querying

+ Writing assertions

Write a real test:

1. Visit a web page.

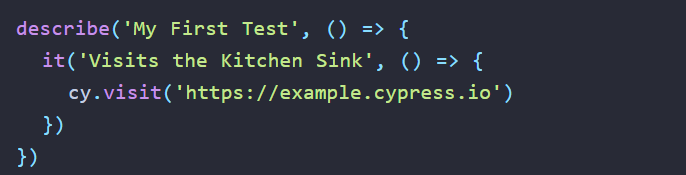
2. Query for an element.

3. Interact with that element.

4. Assert about the content on the page.

- Sau khi đã run app thì chúng ta có thể tiến hành test:

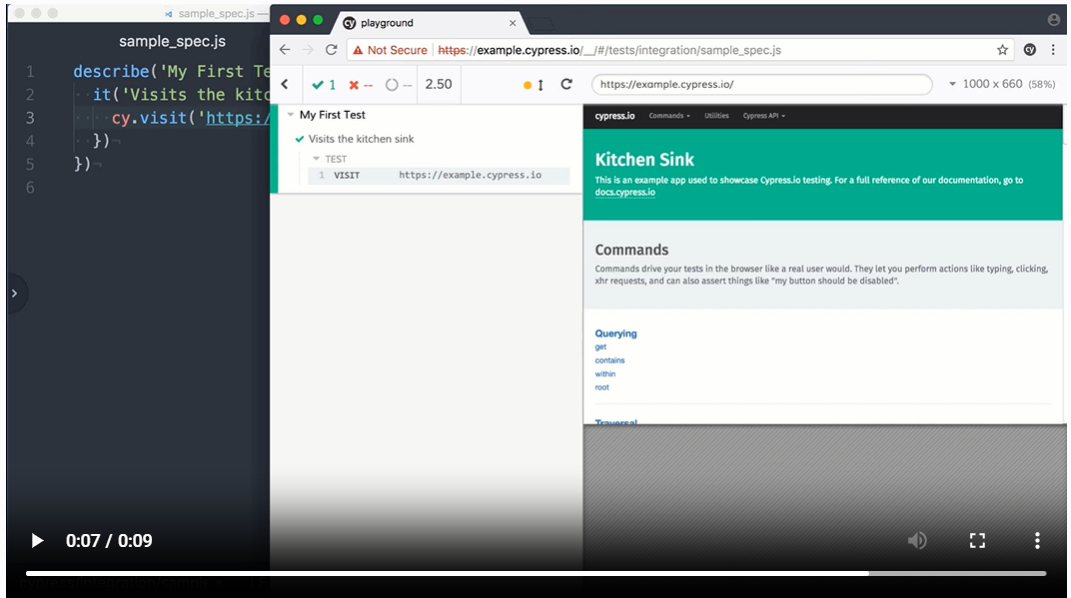
1. Đầu tiên chúng ta có thể test xem có thể truy cập vào url của trang web không, bằng cách truyền url cần test vào trong hàm visit(‘url’)



-> Lưu lại, và quay trở lại Cypress, bạn có thể nhìn thấy sự thay đổi:

+ Xanh -> success

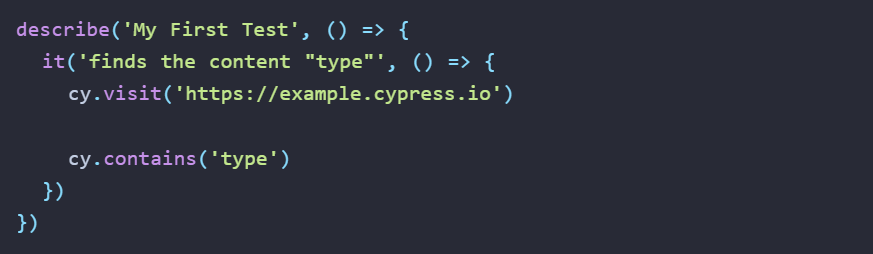
+ Đỏ -> fail



-> Nếu visit() thành công, sẽ thông báo xanh, mặc dù không cần viết assert. Điều này bởi nhiều lệnh của Cypress sẽ tự động được build fail nếu không tìm thấy nội dung mà chúng muốn. Và tự build success nếu tìm thấy -> Cho nên với nhiều lệnh của Cypress, chúng ta không cần viết assert.

1. **Query vào một element:**

- cy.contains(): tìm một element theo nội dung của nó (find by content).



1. **Click an element:**

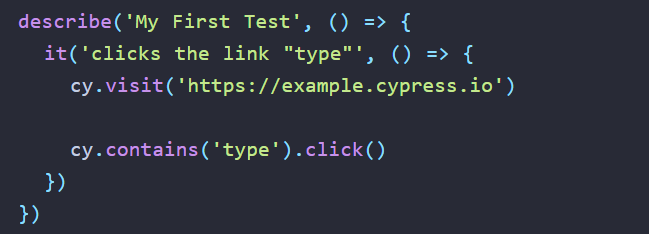
- .click() sẽ giúp chúng ta test cho sự kiện click vào một element

- Chúng ta có thể xâu chuỗi nhiều methods như cách dưới đây:

- VD:

cy.contains(‘…’).click()

… etc



-> Vì theo như ví dụ mẫu này, sau khi link “type” được click vào, sẽ chuyển sang một url mới -> Bởi vậy, công việc test lúc này chính là: chúng ta sẽ đi viết assert kiểm tra xem kết quả có như mong đợi hay không. ( expected == actual ?? )

1. **Assertions:**

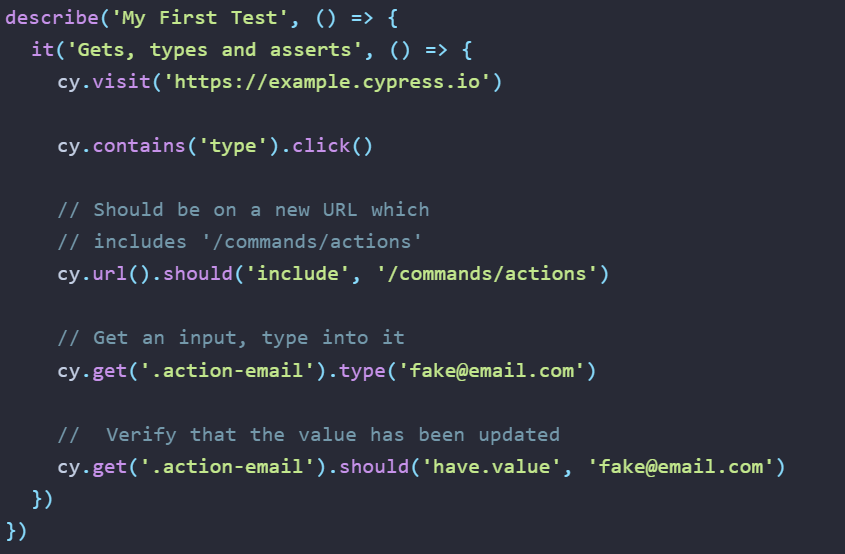
- Điều mà chúng ta có thể nghĩ ngay đến để assert lúc này đó chính là kiểm tra xem url có đúng như mong đợi hay không.

- Cypress cung cấp .should() method để làm việc này.



- Ngoài assert url ở trên ra, chúng ta hoàn toàn có thể kiểm tra thêm một vài thứ khác. (assert không giới hạn)

-> Chúng ta cũng có thể sử dụng cy.get(‘classNameOfElement’) để get ra một element dựa vào class name của nó và kiểm tra (assert) một số thứ.



// nhập dữ liệu vào ô email input (đây là dữ liệu được chuẩn bị, để mình insert vào, rồi assert xem nó có đúng là cái thằng mình vừa insert hay không)

cy.get('.action-email').type('fake@email.com')

*( lệnh .type(‘content’) giúp chúng ta nhập văn bản, value vào trong element )*

// assert kiểm tra

cy.get('.action-email').should('have.value', 'fake@email.com')

\* Lưu ý:

* Không nên xâu chuỗi thêm bất kỳ thứ gì sau một action command (lệnh hành động)
* Không nên get element dựa trên className, tagName, id, textContent. Vì những điều này có thể thay đổi bằng css hoặc js.

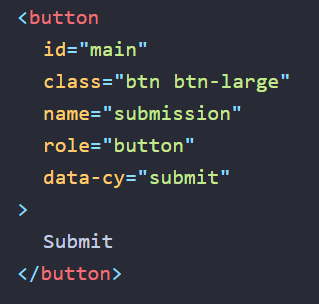
- > Nên get các element bằng cách thêm thuộc tính data-\* vào trong element của bạn, và sử dụng nó cho riêng việc test Cypress.

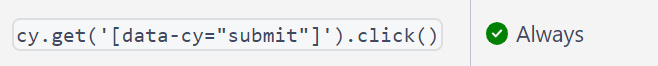
VD: thêm thuộc tính

+ data-cy = “submit”

+ data-tan = “tanTest”

+ …

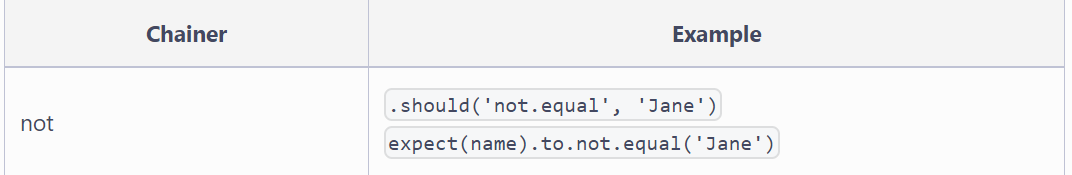




Link tham khảo: *https://docs.cypress.io/guides/references/best-practices#Selecting-Elements*

+ should(chainers, value, method, callbackFn):

Vd:



-> lệnh .should('not.equal', 'Jane') hoàn toàn tương tự như cách chúng ta viết với assert như sau: expect(name).to.not.equal('Jane')

.should('not.equal', 'Jane') = expect(name).to.not.equal('Jane')



+ chai, chai-query, sinon-chai:

Link: *https://docs.cypress.io/guides/references/assertions#Chai*