=====================================================

Tutor: **Rahul Shetty**

Reference: **UDEMY**

Course: **Cypress - Modern Automation Testing from Scratch + Frameworks**

=====================================================

1. Course URL: https://www.udemy.com/course/cypress-tutorial/
2. Document prepared by: **Rajat Verma**
    a. https://www.linkedin.com/in/rajat-v-3b0685128/
    b. https://github.com/rajatt95
    c. https://rajatt95.github.io/

-------------------------------------------------

**Softwares:**

1. Programming language - Javascript
2. Node JS
3. IDE - Visual Studio Code
    a. Plugin
        i. Cucumber (Gherkin) full support
        ii. Excel Viewer

-------------------------------------------------

1. **Learnings from Course (UDEMY - RS - Cypress)**
    a. **Links:**
        i. **Cypress Docs:**
            1. https://docs.cypress.io/guides/guides/launching-browsers#Customize-available-browsers
            2. https://docs.cypress.io/guides/overview/why-cypress
            3. https://docs.cypress.io/api/events/catalog-of-events
            4. https://docs.cypress.io/api/commands/siblings
                a. https://docs.cypress.io/api/commands/next
            5. https://docs.cypress.io/guides/references/configuration
            6. https://docs.cypress.io/guides/tooling/reporters
            7. https://docs.cypress.io/guides/guides/command-line
            8. https://docs.cypress.io/guides/guides/test-retries
            9.
        ii. https://www.npmtrends.com/cypress
        iii. https://nodejs.org/en/download/
        iv. https://code.visualstudio.com/download
        v. https://www.npmjs.com/
        vi. https://www.w3schools.com/jquery/html_removeattr.asp

vii. https://developer.mozilla.org/en-US/docs/Web/JavaScript

**viii. Mochawesome Reports**

1. https://www.npmjs.com/package/mochawesome
2. https://www.npmjs.com/package/cypress-mochawesome-reporter

ix. **Cucumber**:

1. **DataTables**:
    a. https://github.com/cucumber/cucumber-js/blob/main/features/data_tables.feature
2. **Reports**:
    a. https://github.com/wswebcreation/multiple-cucumber-html-reporter
    b. Add screenshots on failure:
        i. https://github.com/qaboxletstest/cypress-cucumber-demo/blob/master/README.md
        **ii. https://github.com/dane-harnett/cypress-cucumber-attach-screenshots-to-failed-steps**

x. Intercept:

1. https://docs.cypress.io/api/commands/intercept

xi. API testing:

1. https://docs.cypress.io/api/commands/request
2. https://docs.cypress.io/api/commands/request#Assertions

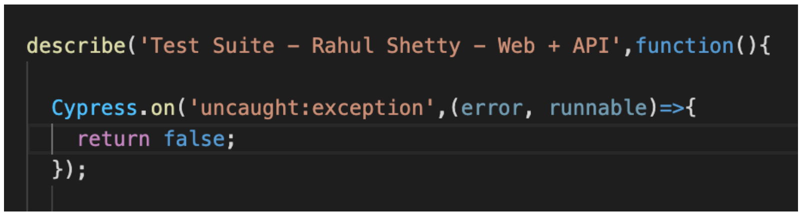
xii. SSO:

1. https://docs.cypress.io/guides/guides/web-security#Same-superdomain-per-test

xiii. Rahul Shetty:

1. https://rahulshettyacademy.com/
2. https://rahulshettyacademy.com/#/practice-project
3. https://rahulshettyacademy.com/seleniumPractise/
4. https://rahulshettyacademy.com/AutomationPractice/
5. https://rahulshettyacademy.com/angularpractice/
6. https://rahulshettyacademy.com/angularAppdemo/

xiv. **Catch Uncaught exception**:

1. 

```
describe('Test Suite - Rahul Shetty - Web + API',function(){

  Cypress.on('uncaught:exception',(error, runnable)=>{
    return false;
  });
```

**xv. Web+API:**

1. ***Skip login***
    a. Call Login API
    b. Extract token
    c. Inject into Browser Local storage

1. Same clas
2. Different class
    a. Export the class
    b. Import it and create the object of that class

x. OOPS
    1. Inheritance

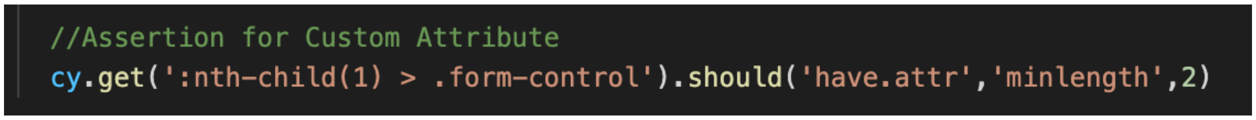
c. **Cypress**
    i. Architecture
    ii. Browser support
        1. Chrome
        2. Electron
        3. Firefox
    iii. Components
        1. Test Runner
            a. **node_modules/.bin/cypress open**
        2. Dashboard service
    iv. Plugin
        1. Frame
            **a. npm install -D cypress-iframe**
    v. Locator Strategy
        1. CSS Selector
    vi. Stages:
        1. Pending
        2. Resolved
        3. Rejected
    vii. Click using
        1. Element text
            a. cy.**contains**('PROCEED TO CHECKOUT').**click**()
        2. CSS selector
            a. cy.**get**('.cart-icon > img').**click**()
    viii. Dropdown
        1. Static
        2. Dynamic
    ix. Radio button
    x. Alerts/Pop-ups
        1. Cypress Auto-Accepts
        2. Cypress has capabilities to listen Browser events
            a. https://docs.cypress.io/api/events/catalog-of-events
            b. Get the text and validate the text
        3. Events:
            a. window:alert
            b. window:confirm
    xi. How to handle Child tabs

xii. Navigating Browser controls
xiii. Get current URL
xiv. Handle Web Tables
xv. Mouse Hover
xvi. Click on Hidden Element
xvii. How to grab the attribute value
xviii. Handle Frames
xix. Override the behavior of Cypress.json
xx. **Scripting commands in Package.json file for CI Integration**
**xxi. Running Multiple specs file on fly from Cypress Scripting commands**
xxii. Cypress.**config**()
xxiii. Operations/Functions

1. cy.**visit**('https://www.google.com/') -> To navigate to URL
2. cy.**type**('Hello, Test Automation Engineer') -> To type something in 'textbox
3. cy.**wait**(3000) -> Will wait for 3 seconds
4. cy.**get**('.product:visible').**should**('have.length',4) ->
   a. Will return only visible elements
   b. Assertion for count 4 web elements
5. cy.**get**('.products').**find**('.product') ->
   a. Cypress will look for elements only inside the web element which has class '.products'
6. cy.**get**('.products').**find**('.product').**eq**(2).**contains**('ADD TO CART').**click**() ->
   a. Out of 4 elements -> Go to 2nd element which has text 'ADD TO CART' and perform click operation on it.
7. cy.**get**('.products').**then()**
8. cy.**log**("elementLogo.text(): "+elementLogo.text()) ->
   a. This will add the details in Cypress steps
9. Aliasing -> To re-use Locators
   cy.**get**('.products').**as**('**productLocator**')
   cy.**get**('**@productLocator**').**find**('.product').**should**('have.length',4)
10. cy.**get**('select#dropdown-class-example').**select**('option2').**should**('have.value','option2')
11. cy.**on**('**window:alert**',(str) => {
    //Two Strings comparison using Mocha framework
    **expect**(str).to.equal('Hello , share this practice page and share your knowledge')
    })
12. cy.get('#opentab').**invoke**('**removeAttr**','target').click()
13. cy.**go**('back')
    cy.**go**('forward')
14. cy.**url**()

15. cy.get('.mouse-hover-content').**invoke**('show')
16. cy.**frameLoaded**('#courses-iframe')
17. cy.**iframe**().find("a[href='#/mentorship']").eq(0).click()
18. cy.**pause**()
19. cy.**debug**()
20. jQuery method:
    a. cy.get('#opentab').then(function(e1){
       //Getting the attribute 'href' value
       const attribute_href = e1.**prop**('href')
       cy.log("Attribute - href value is : "+attribute_href)
       cy.visit(attribute_href)
       })

d. **Assertions** (using the **Mocha** framework):
   i. Elements count
   ii. Element Text
   iii. Checkbox
      1. should be enabled/checked
      2. should not be enabled/checked
   iv. Multiple assertions in a single line
   v. Attribute validation



```
//Assertion for Custom Attribute
cy.get(':nth-child(1) > .form-control').should('have.attr','minlength',2)
```

   vi. Dropdown value
   vii. Visible and Invisible elements
   viii. Radio button
   ix. Two Strings
      1. Should have
   x. Substring assertion
      1. Should include

e. Create new project
   i. package.json
      1. **npm -i init**
      2. With default values, this package.json file is created
   ii. **npm install cypress --save-dev**
   iii. Cypress:
      1. **cy** -> Similar to the driver in Selenium WebDriver
         a. This cy will be used to perform all operations over the Browser

f. Execution using command line/Terminal
   i. Headed mode
   ii. Headless mode

g. Each **.js file** is called a **spec file** in Javascript terminology
h. Project framework Structure
i. Override the default behavior of Cypress using the cypress.json file
**j. UI Test Automation Framework**
   i. Design Pattern - Page Object Model
      1. Using **./cypress/support/pageObjects/HomePage.js**
         a. Export and Import
         b. Object creation
   ii. Data-Driven approach
      1. Using **./cypress/fixtures/example.json**
   iii. Reporting
      1. Cypress dashboard
      2. Mocha Awesome Reports
   iv. CI/CD
      1. Jenkins
   v. Setup of Hooks
      1. Before and After Test
   vi. Utilities:
      1. Custom Cypress Commands - Using **./support/commands.js**
      2. Re-try failed test cases
      3. Screenshots on failure
      4. Videos for test execution
      **5. Parallel execution?**
   vii. Override the behavior of Cypress.json
      1. Cypress.config()
   viii. Run from cmd prompt/Terminal
      1. **node_modules/.bin/cypress run --spec cypress/integration/3-RS_AngularPractice/_14_Cypress_EnvVariable_FromCypressJSON.js --env application_URL=https://www.google.com/ --browser firefox --headed**
   **ix.**

**k. Cucumber**
   i. https://cucumber.io/
   ii. https://cucumber.io/docs/installation/javascript/
   iii. https://github.com/cucumber/cucumber-js
   iv. https://github.com/TheBrainFamily/cypress-cucumber-example
   v. https://github.com/TheBrainFamily/cypress-cucumber-preprocessor
   vi. Tagging in features
      1. @focus, @sanity, @bvt
   vii. Run all or specific features
   **viii. Multiple-cucumber-html-reporter**

ix. **Data-Driven in Cucumber:**
   1. https://github.com/TheBrainFamily/cypress-cucumber-preprocessor/blob/master/cypress/integration/ScenarioOutline.feature

l. **Mocking HTTP requests/responses with Cypress (XHR Testing)**
   i. https://docs.cypress.io/api/commands/intercept
   ii. https://docs.cypress.io/api/commands/intercept#Request-object-properties
   **iii. We can**
      1. Modify Real HTTP Response
      2. Change the Response Body
      3. Headers
      4. HTTP status codes
         a. Before being received by the Browser
   **iv. We can**
      1. Modify the HTTP request's body
      2. Request Headers
      3. Request URLs
         a. Before sending it to the server
   **v. Cypress helps us to perform the Integration testing between UI and back-end services**
   vi. **API testing:**
      1. https://docs.cypress.io/api/commands/request
      2. https://docs.cypress.io/api/commands/request#Assertions

m. **Single Sign-on ( SSO) Automation Testing with Cypress**
   i. https://docs.cypress.io/guides/guides/web-security#Same-superdomain-per-test

n. **Session Token & Local Storage Data saving with Cypress & CSV Parsers**

o. **Cypress DB Integration Testing Strategy**

**================================================================**

## Other than course (Rajat):

**i. Delete directory before execution:**

**ii. Allure reports**
   1. https://www.youtube.com/watch?v=eDW6BW2cflA&ab_channel=QABoxLet%27sTest
   2. https://www.npmjs.com/package/%40shelex/cypress-allure-plugin

iii. **Cypress XPath**
   1. https://www.youtube.com/watch?v=YV3qPvhJ-rg&ab_channel=remarkablemark
   2. Tagging via Cypress.json file

iv. **Screenshots on failure of test case (BDD) (NW)**
   1. https://github.com/qaboxletstest/cypress-cucumber-demo/blob/master/README.md
   2. https://github.com/dane-harnett/cypress-cucumber-attach-screenshots-to-failed-steps

v. **Mochawesome reports options**
   1. https://www.lambdatest.com/blog/how-to-generate-mocha-reports-with-mochawesome/

vi. **Browserstack integration**
   1. https://www.browserstack.com/docs/automate/cypress
   2. https://www.browserstack.com/docs/automate/cypress/cli-reference#run-tests

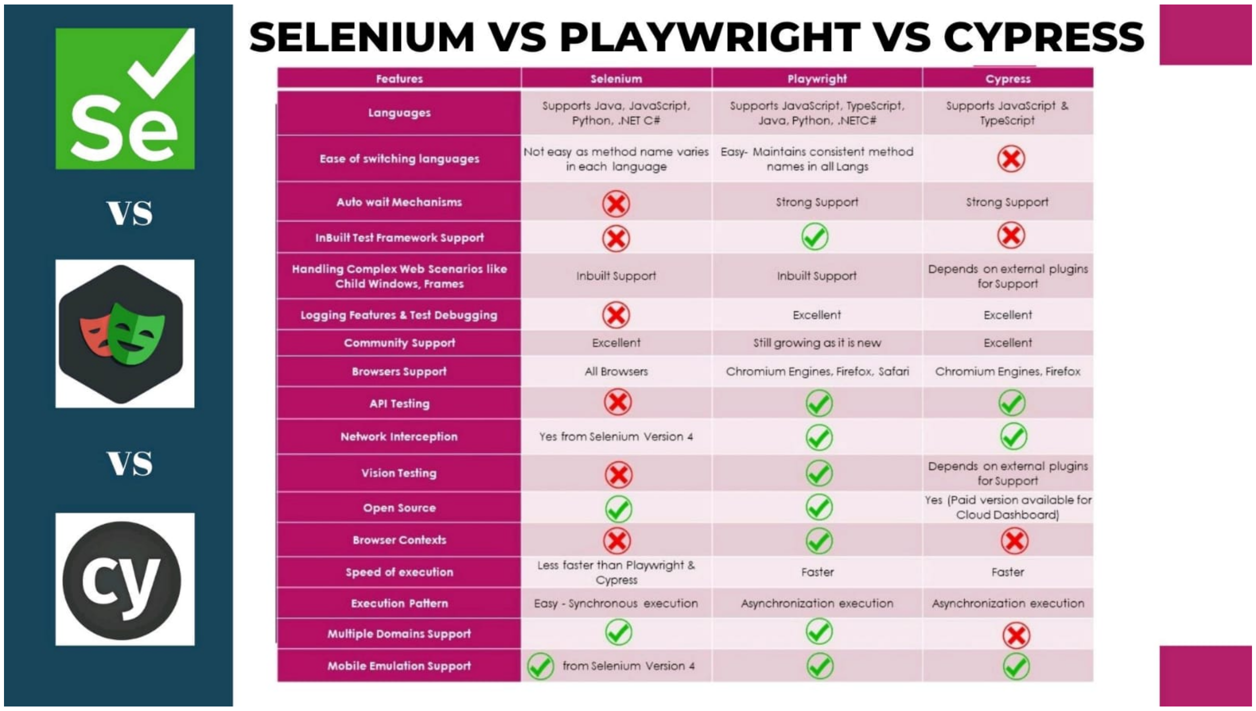
vii. **CircleCI integration (NW)**
   1. https://circleci.com/developer/orbs/orb/cypress-io/cypress
   2. https://kailash-pathak.medium.com/cypress-test-case-execution-in-ci-cd-using-circleci-fab21028a169

viii. **Docker integration (NW)**
   1. https://www.youtube.com/watch?v=h8wd0V0Yes8&ab_channel=TheTestingAcademy

ix. **How to save the Login tokens in browser cookies using Cypress**
   1. https://stackoverflow.com/questions/70554024/in-cypress-how-to-setcookie-before-test/70554079#70554079

=====================================================================

=====================================================================



## SELENIUM VS PLAYWRIGHT VS CYPRESS

| Features | Selenium | Playwright | Cypress |
|---|---|---|---|
| Languages | Supports Java, JavaScript, Python, .NET C# | Supports JavaScript, TypeScript, Java, Python, .NETC# | Supports JavaScript & TypeScript |
| Ease of switching languages | Not easy as method name varies in each language | Easy- Maintains consistent method names in all Langs | ✗ |
| Auto wait Mechanisms | ✗ | Strong Support | Strong Support |
| InBuilt Test Framework Support | ✗ | ✓ | ✗ |
| Handling Complex Web Scenarios like Child Windows, Frames | Inbuilt Support | Inbuilt Support | Depends on external plugins for Support |
| Logging Features & Test Debugging | ✗ | Excellent | Excellent |
| Community Support | Excellent | Still growing as it is new | Excellent |
| Browsers Support | All Browsers | Chromium Engines, Firefox, Safari | Chromium Engines, Firefox |
| API Testing | ✗ | ✓ | ✓ |
| Network Interception | Yes from Selenium Version 4 | ✓ | ✓ |
| Vision Testing | ✗ | ✓ | Depends on external plugins for Support |
| Open Source | ✓ | ✓ | Yes (Paid version available for Cloud Dashboard) |
| Browser Contexts | ✗ | ✓ | ✗ |
| Speed of execution | Less faster than Playwright & Cypress | Faster | Faster |
| Execution Pattern | Easy - Synchronous execution | Asynchronization execution | Asynchronization execution |
| Multiple Domains Support | ✓ | ✓ | ✗ |
| Mobile Emulation Support | ✓ from Selenium Version 4 | ✓ | ✓ |

==============================================================