Cypress Basics

Cypress is the open-source and free test automation tool. It can be used for front end and application programming interface (API) test automation. This tool is mainly developed to solve the issues that the teams face, while automating an application.Any application which runs on browser can be automated. Testing of modern web applications.Doesnot use any selenium.

Cypress helps to achieve the following −

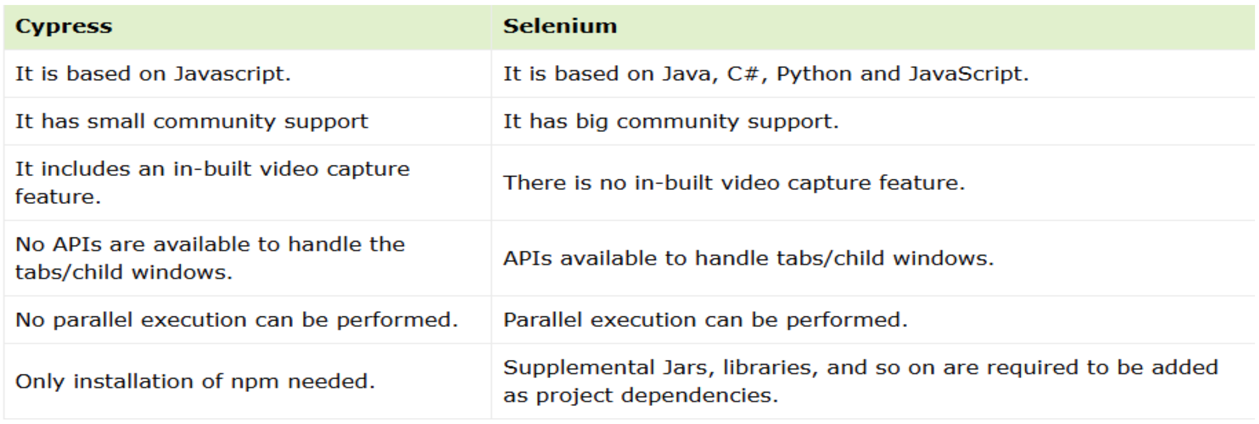
a)Configure tests.
b)Execute tests.
c)Identify errors (if any).

Key Features of Cypress:

1)Time Travel: Cypress captures snapshots as your tests run, enabling you to go back in time and see what happened at each step.
2)Real-Time Reloads: As you save your test files, Cypress automatically reloads them, showing the changes instantly.
3)Automatic Waiting: Cypress waits for commands and assertions to pass without the need for explicit waits (cy.wait()).
4)Network Traffic Control: You can stub network responses, check for network errors, and control network requests.
5)Easy Debugging: With detailed error messages and stack traces, Cypress helps in quickly identifying issues.

Disadvantages

1)It is only based on JavaScript.
2)A relatively new tool and hence, the community support is not extensive.
3)It cannot perform mobile testing.



Cypress	Selenium
It is based on Javascript.	It is based on Java, C#, Python and JavaScript.
It has small community support	It has big community support.
It includes an in-built video capture feature.	There is no in-built video capture feature.
No APIs are available to handle the tabs/child windows.	APIs available to handle tabs/child windows.
No parallel execution can be performed.	Parallel execution can be performed.
Only installation of npm needed.	Supplemental Jars, libraries, and so on are required to be added as project dependencies.

Cypress Environment Setup

Step 1: Install Node.js and NPM

Make sure you have Node.js installed. You can download it from Node.js official site. Installing Node.js will also install npm (Node Package Manager).

Step 2: Initialize a New Project

If you don't have a project directory yet, create one and initialize it:

```
mkdir cypress-test
cd cypress-test
npm init -y
```

Step 3: Install Cypress

Install Cypress as a development dependency:

```
npm install cypress@latest --save-dev  or
npm install cypress --save-dev
```

Step 4: Open Cypress for the First Time

Run Cypress to set it up and generate the initial folder structure:

```
npx cypress open
```

This will create a cypress/ folder with the following structure:

```
cypress/
├── e2e/
├── fixtures/
├── support/
cypress.config.js
```

Step 5: Create a Cypress Test File

Inside the cypress/e2e/ folder, create a new test file named inputFieldTest.cy.js.

```
touch cypress/e2e/inputFieldTest.cy.js
```

Step 6: Write Your Cypress Test

Paste the following test code in the newly created file:

```
describe('Input Field Testing', () => {
  beforeEach(() => {
     // Visit the specified URL
     cy.visit('http://surajtest.atspace.cc/cypress.php');
  });

  it('should verify default value and type into the input field', () => {
     // Check if the input field exists
     cy.get('#username').should('exist');

     // Verify the default value of the input field
     cy.get('#username').should('have.value', 'testusername');

     // Clear the existing value and type a new value into the input field
     cy.get('#username').clear().type('testuser');

     // Verify the new value has been entered correctly
     cy.get('#username').should('have.value', 'testuser');

     // Click the submit button
     cy.get('#submit-btn').click();

     // Check for any feedback or confirm action (if implemented)
     cy.get('#feedback').should('exist');
  });
});
```

Step 7: Configure Cypress (Optional)

If you need to adjust the base URL or other configurations, you can edit the cypress.config.js file:

```
const { defineConfig } = require('cypress');
```

```
module.exports = defineConfig({
  e2e: {
    baseUrl: 'http://surajtest.atspace.cc',
    setupNodeEvents(on, config) {
      // implement node event listeners here
    },
  },
});
```

This way, you can use cy.visit('/cypress.php') instead of the full URL in your tests.

Step 8: Run the Cypress Test

Open Cypress with:

```
npx cypress open
```

In the Cypress UI:

Select E2E Testing mode.

Choose a browser (e.g., Chrome).

Click on inputFieldTest.cy.js to run the test.

Alternatively, you can run it headlessly in the terminal:

```
npx cypress run
```

Step 9: Check the Results

Cypress will show a visual representation of the test running if you use the UI.

In headless mode, you'll see the test results in your terminal.

Youtube:

https://www.youtube.com/watch?v=cnnkb0AuIFI&list=PLUDwpEzHYYLvA7QFkC1C0y0pDPqYS56iU

Linkedin:
https://www.linkedin.com/feed/update/urn:li:activity:7105488699827654656/