**Playwright**

Npm init 🡪 creates package.json

npm install playwright chai @cucumber/cucumber cucumber-html-reporter prettier

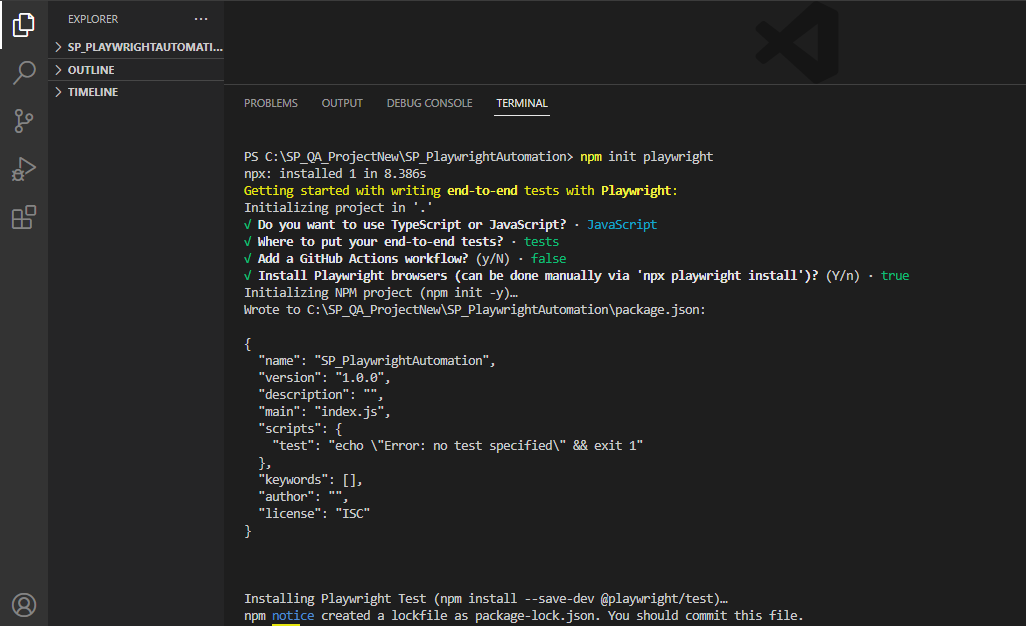
# **Non BDD:**

## Installation:

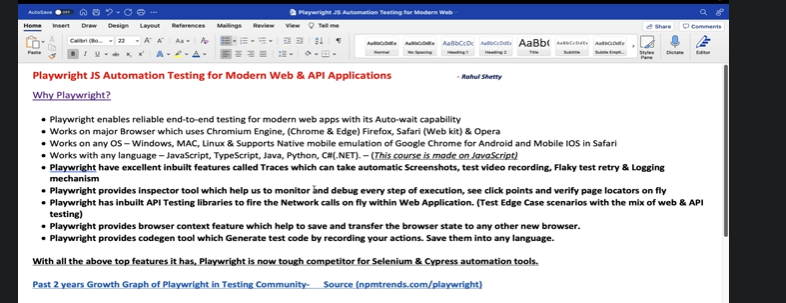
Make sure node.js installed in your system and you provide node.js path in your system variable

Node version 16 and npm 9

Npm init playwright



Features of playwright



Async and Await Understanding

Async and await comes together🡪 async for function

Await is to wait until the particular line of code executes completely

{browser} - >To recognize as playwright fixture (Global variable, so it’s different from browser argument to function)

Npx playwright test

Under playwright config json file make use of ‘use’ property to specify browser type

 use: {

   browserName : 'chromium',

   headless : true

  },

await page.locator('#password').type('learing');

await page. ocator('#signInBtn').click();

await expect(page.locator("[style\*='block']")).toContainText('Incorrect username/password.')

await page.locator('#username').fill('');  // to clear the populated text from field

Multiple webelements :

await page.locator(‘.card-body a’).first().textContent()

await page.locator(‘.card-body a’).nth(1).textContent()

List of webelements

const cardTitles = await page.locator(‘.card-body a’)

const allTitles = await cardTitles.allTextContents();

It return an array 🡪 [ ‘Samsung’, ‘Iphone’, ‘Nokia’ ]

Wait dynamically for new page in service based applications:

Await page.waitForLoadState(‘networkidle’)

Await promise.all( [

Page.waitForNavigation(),

signIn.click(),

]);

Dropdown—static

const dropdown = page.locator(‘select.countrydropdown’);

await dropdown.selectOption(‘India’);

Selecting a radio button and checkbox

await page.locator(‘.checkboxName’);.toBeChecked()

await page.locator(‘.namerbtn’).isChecked()

expect (**await** page.locator(‘.namerbtn’).isChecked()).toBeFalsy() // toBeTruthy();

.uncheck()

**Await** expect (page.locator(‘.namerbtn’).last()).toBeChecked()).

Blinking text validations using

<body>

<a href=’www.google.come’ class=’blinkingText’

await expect(page.locator(‘[href\*=’google’).toHaveAttribute(‘class’, ‘blinkingText’);

Child windows and tabs handling:

In the same tab:

And in the new separate tab

const context = await browser.newContext();

Const newPage = Await promise.all([

context.waitForEvent(‘Page’); // before event occurs at below click link ,wait

elementLink.click();

])

newPage.locator(‘.read).click();

const products= await page.locator(‘.body\_cards’);

const count =products.count();

for (let i=0; i<count; ++i){

products.nth(i); //supports chaining🡪 products.nth(i).locator(‘insideProductItemLocator’).textContent()

}

Text based locator

Auto suggestive placeholder dropdown handling [type searching in drop down]

await page.locator(‘.dropdown’).type(‘Indi’, {delay :100});

Make sure options section open

Debug mode:

npx playwright test pageObj1.spec.js –debug

Must watch for api testing

<https://www.youtube.com/watch?v=-pH6Dax-OyY>

<https://www.youtube.com/watch?v=hk6ND5gVdyc>

Test log and readability / soft and hard assertions

<https://www.youtube.com/watch?v=Czroj_xzOnY>

<https://www.youtube.com/watch?v=ZvZ2C-JTyWo>

Implementing Parameterization in running tests with different data sets

How to pass test data as fixture by extend test annotation behavior

How to drive the data from external json files to playwright test

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

We can have multiple custom config files

To run tests against config files🡪

But make sure you are at project level ,instead of tests folder level in terminal

npx playwright test tests/parameterized.spec.js --config playwright.config2.js

npx playwright test tests/parameterized.spec.js --config playwright.config3.js

npx playwright test tests/parameterized.spec.js --config playwright.config4.js



Parallel and serial

By default test files/specs will trigger parallel

// individual tests in file will run in sequence

How to run tests in file run in parallel

Test.describe.configure({mode:’parallel’});

Test(‘ ‘ =>{

}

Test(‘ ‘ =>{

}

Test(‘ ‘ =>{

}

How to run tests in file run in serial

How to run tests in file run in parallel

Test.describe.configure({mode:’serial’});

Test(‘ ‘ =>{

}

Test(‘ ‘ =>{

}

Test(‘ ‘ =>{

}

How to tag tests and control test execution using command line parameters:

Npx playwright test --grep @smoke

Test(‘@smoke ‘ =>{

}

Test(‘@regression @smoke ‘ =>{

}

Test(‘@regression ‘ =>{

}

Reports:

npm i -D @playwright/test allure-playwright

Npx playwright test --grep @smoke –reporter=line,allure-playwright

allure generate ./allure-results –clean

allure open ./allure-report

How to create a custom scripts in package.json and trigger from command line

The same we did it in cypress:

|  |
| --- |
| }, |
|  | "scripts": { |
|  | "clean:reports":"(if exist cypress\\reports (rmdir /Q /S cypress\\reports)) && mkdir cypress\\reports && mkdir cypress\\reports\\mocha && mkdir cypress\\reports\\mochareports", |
|  | "pretest": "npm run clean:reports", |
|  | "scripts": "cypress run --spec cypress/integration/DemoTests/\*\*/\*", |
|  | "combine-reports": "mochawesome-merge cypress/reports/mocha/\*.json > cypress/reports/mochareports/report.json", |
|  | "generate-report": "marge cypress/reports/mochareports/report.json -f report -o cypress/reports/mochareports", |
|  | "posttest": "npm run combine-reports && npm run generate-report", |
|  | "test": "npm run scripts || npm run posttest" |

Handle multiple environments in Playwright

Playwright Framework #2 | Execute in multiple environment | Playwright Tutorial - Part 80

<https://www.youtube.com/watch?v=GQStVI5qbLI>

**Handling child window:**

Test(‘child window handling’, ({browser}) = > {

Const context = await browser.newContext();

Const page = await context.newPage();

Await page.goto(‘https/cnncm.com/sjs’);

Const docLink = page.locator(‘href\*=’documents\_link’);

Const [newPage]= await Promise.all([

Context.waitForEvent(‘page’),

docLink.click(),

]);

Text = await newPage.locator(‘.name).textContent();

Console.log(text);

//switching back to Parent window

Await Page.locator(‘#username’).type(‘Sushant’);

}

Frames handling: --Use selectorsHub chrome plugin

Const newFrame= Page.frameLocator(“#courses-iframe”);

newFrame.locatore(‘.btn).click();