Pasos para configurar un proyecto de cypress usando POM integrado con Cucumber

Javier Flores

www.linkedin.com/in/javier-flores-borrego

1. Crea la carpeta con el nombre de la web que vas a testear donde quieras ubicar el proyecto, abrela con VS code (o el editor/IDE que prefieras), abre la consola y escribe:

- npm init -y

Esto inicializa un nuevo proyecto Node.js y crea un archivo package.json con la configuración predeterminada

2. Instala Cypress y estos 2 plugin escribiendo en la consola

npm install --save-dev cypress @badeball/cypress-cucumber-preprocessor @bahmutov/cypress-esbuild-preprocessor esbuild

3. Abre cypress y sigue todos los pasos para que cree los archivos de configuración

npx cypress open

4. Dentro de la carpeta e2e crea una carpeta para los step definitions, y llámala “step-definitions”, dentro de esta carpeta crearemos las diferentes archivos .js para organizar nuestros steps

5. En el archivo package.json añade esto:

```
"cypress-cucumber-preprocessor": {
  "stepDefinitions": "cypress/e2e/step-definitions/*.js"
},
```

(si le has puesto otro nombre a la carpeta de los steps o la has colocado en otro lugar recuerda cambiar la ruta)

6. En el cypress.config.js sustituye lo que tengas por esto

```
const { addCucumberPreprocessorPlugin } =
require('@badeball/cypress-cucumber-preprocessor');
const { createEsbuildPlugin } =
require('@badeball/cypress-cucumber-preprocessor/esbuild');
const createBundler =
require('@bahmutov/cypress-esbuild-preprocessor');
const { defineConfig } = require('cypress');

module.exports = defineConfig({
  e2e: {
    "chromeWebSecurity": false,
    specPattern: '**/*.feature',
    async setupNodeEvents(on, config) {
      await addCucumberPreprocessorPlugin(on, config);
      on(
        'file:preprocessor',
        createBundler({ plugins: [createEsbuildPlugin(config)] })
      );
      return config;
    }
  }
})
```

7. Ve al icono de extensiones de VS code, busca e Instala la extensión: Cucumber (Gherkin) Full Support y después ve a las settings.json en .vscode y añade esto

```
{

    "cucumberautocomplete.steps": [
    "/cypress/e2e/step-definitions/*.js"],
    },
}
```

Al hacer esto podrás visitar los steps desde los archivos .feature y se remarcarán los steps que no existan (deberás recargar VS code para que funcione correctamente)

8. Crea una carpeta para las clases llamada “pages”

9. Dentro crea 1 archivo llamado commonPage.js (después crearemos un archivo para cada “página”) desde donde exportar las clases de esta forma:

```
export class CommonPage{
  visitLink(url) {
    cy.visit(url);
  }
}
```

En estos archivos estarán nuestras funciones que luego serán usadas en los steps

9. En la carpeta de los steps crea un archivo llamado commonSteps.js (después crearemos un archivos .js por cada página) en ellos estarán alojados los steps que usaremos en nuestros tests, para que estos archivos funcionen correctamente necesitamos :

- Importa las palabras claves When, then…
- Importar la/s clase/s que vayamos a necesitar
- Declarar las instancias de cada clase

```
import { When, Then, Given } from
"@badeball/cypress-cucumber-preprocessor";

// Clases importadas
import { CommonPage} from "../Pages/commonPage"

//Instancias de clase
let commonPage = new CommonPage();
```

Los steps se componen de varias partes:

- Partícula de gherkin: Given, When, Then, And
- Texto del step, en caso de que haya algún parámetro se indica entre llaves {} y se indica su tipo, string (texto)

```
Given("I visit {string}", (url) => {
  commonPage.visitLink(url);
});
```

- int (numero)

```
Given("the list have {int} elements", (elementNumber) => {
  mainPage.listElementNumber(elementNumber);
});
```

Estos son los que más se usan, aunque también podría ser un booleano, objeto o array.

10 Crea el primer test en un archivo loginTest.feature de esta forma:

```
#Para comentar en un archivo .feature se hace con este símbolo #

#Esto es la descripción de la batería de test contenida en este archivo
Feature: Login test suite

Background:
#Esto es equivalente al beforeEach
  Given I visit "https://www.saucedemo.com/"

#Los Scenarios son los tests (lo que antes era "it")
  Scenario: login succesfull
  Given I visit "https://www.saucedemo.com/"
```