Tutorial: Pruebas Selenium en .NET con NUnit sobre Wikipedia

1️⃣ Crear la solución y proyecto

Abrimos la terminal:

mkdir SeleniumWikipedia

cd SeleniumWikipedia

dotnet new sln -n SeleniumWikipedia

1.1 Crear proyecto de pruebas NUnit

dotnet new nunit -n WikipediaTests

dotnet sln add WikipediaTests/WikipediaTests.csproj

1.2 Instalar paquetes necesarios

cd WikipediaTests

dotnet add package Selenium.WebDriver

dotnet add package Selenium.WebDriver.ChromeDriver

dotnet add package Selenium.Support

cd ..

Explicación:

- Selenium.WebDriver → Librería principal para interactuar con el navegador.
- Selenium.WebDriver.ChromeDriver → Driver para Chrome.
- Selenium.Support → Clases de soporte adicionales para Selenium.

2️⃣ Código de prueba Selenium

Archivo: WikipediaTests/WikipediaTests.cs

```
using NUnit.Framework;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;

namespace WikipediaTests
{
  public class WikipediaHomeTests
  {
    private IWebDriver driver;
```

```
[SetUp]
public void Setup()
{
   var options = new ChromeOptions();
   options.AddArgument("--headless"); // Ejecutar sin abrir ventana
   options.AddArgument("--no-sandbox");
   options.AddArgument("--disable-dev-shm-usage");
   driver = new ChromeDriver(options);
}

[Test]
public void VerificarTituloPaginaPrincipal()
{
   driver.Navigate().GoToUrl("https://www.wikipedia.org/");
   string titulo = driver.Title;
   Assert.AreEqual("Wikipedia", titulo);
}

[Test]
public void VerificarCajaBusquedaExiste()
{
   driver.Navigate().GoToUrl("https://www.wikipedia.org/");
   var cajaBusqueda = driver.FindElement(By.Id("searchInput"));
   Assert.IsTrue(cajaBusqueda.Displayed);
}

[Test]
public void BuscarPalabraYVerificarResultado()
{
   driver.Navigate().GoToUrl("https://www.wikipedia.org/");
```

```
        var cajaBusqueda = driver.FindElement(By.Id("searchInput"));
        cajaBusqueda.SendKeys("Selenium (software)");
        cajaBusqueda.Submit();

        var header = driver.FindElement(By.Id("firstHeading"));
        Assert.AreEqual("Selenium (software)", header.Text);
    }

    [TearDown]
    public void Teardown()
    {
        driver.Quit();
    }
  }
}
```

✅ **Explicación detallada**

1. **Setup**
    - ChromeOptions con --headless permite ejecutar Chrome sin abrir ventana.
    - --no-sandbox y --disable-dev-shm-usage son necesarios en entornos Linux como GitHub Actions.
2. **Test: VerificarTituloPaginaPrincipal**
    - Navega a Wikipedia y verifica que el título sea "Wikipedia".
3. **Test: VerificarCajaBusquedaExiste**
    - Comprueba que la caja de búsqueda principal existe y es visible.
4. **Test: BuscarPalabraYVerificarResultado**
    - Escribe "Selenium (software)" en la caja de búsqueda, envía el formulario y verifica que la cabecera de la página de resultados coincide.
5. **Teardown**
    - Cierra Chrome al finalizar cada prueba con driver.Quit().

3️⃣ **Configurar GitHub Actions para Selenium**

Archivo: .github/workflows/selenium-ci.yml

```
name: Selenium CI

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  selenium-tests:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v3

      - name: Setup .NET
        uses: actions/setup-dotnet@v3
        with:
          dotnet-version: '7.0.x'

      - name: Install Chrome
        run: |
          sudo apt-get update
          sudo apt-get install -y wget unzip xvfb libxi6 libgconf-2-4
          wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
          sudo dpkg -i google-chrome-stable_current_amd64.deb || sudo apt-get -f install -y

      - name: Restore dependencies
        run: dotnet restore WikipediaTests/WikipediaTests.csproj
```

```
- name: Run Selenium Tests
  run: dotnet test WikipediaTests/WikipediaTests.csproj --configuration Release
```

✅ **Explicación paso a paso**

1. **runs-on: ubuntu-latest** → Ejecuta en un runner Linux de GitHub.
2. **Install Chrome** → Necesario para Selenium. Incluye dependencias y descarga de Chrome estable.
3. **dotnet restore** → Descarga paquetes NuGet (Selenium y NUnit).
4. **dotnet test** → Ejecuta todas las pruebas headless.

Importante: --headless permite que Chrome corra sin abrir ventana, lo que es obligatorio en GitHub Actions.

🚀 **Ejecutar el flujo CI/CD**

1. Haz commit y push del proyecto completo a GitHub.
2. GitHub Actions detectará cambios en la rama main.
3. Verás un job llamado **Selenium CI** ejecutando las pruebas.
4. Todas las pruebas se ejecutarán de forma automática en el runner, incluyendo la búsqueda en Wikipedia.