Microsoft

Workshop CI Integration for Playwright Tests

Module 4

Conditions and terms of use

Learning Units covered in this Module

- Continuous Integration
- Steps to run tests in CI
- GitHub Actions
- Azure pipelines

Continuous Integration

Continuous Integration

CI involved the automatic execution of test scripts whenever a developer commits a new code aims to validate the correctness of the changes and maintain overall stability of the application.

Advantages of Using Continuous Integration (CI) for Test Automation:

- Early Defect Detection
- Integration Testing
- Consistent Test Execution
- Continuous validation
- Increases Tests coverage
- Increases Developer Productivity
- Improves the quality assurance process and ensures the timely release of high-quality software.

Playwright tests can be run on any CI provider.

Steps:

1. Ensure CI agent can run browsers

2. Install Playwright

```
# Install NPM packages
npm ci

# Install Playwright browsers and dependencies
npx playwright install --with-deps
```

3. Run your tests

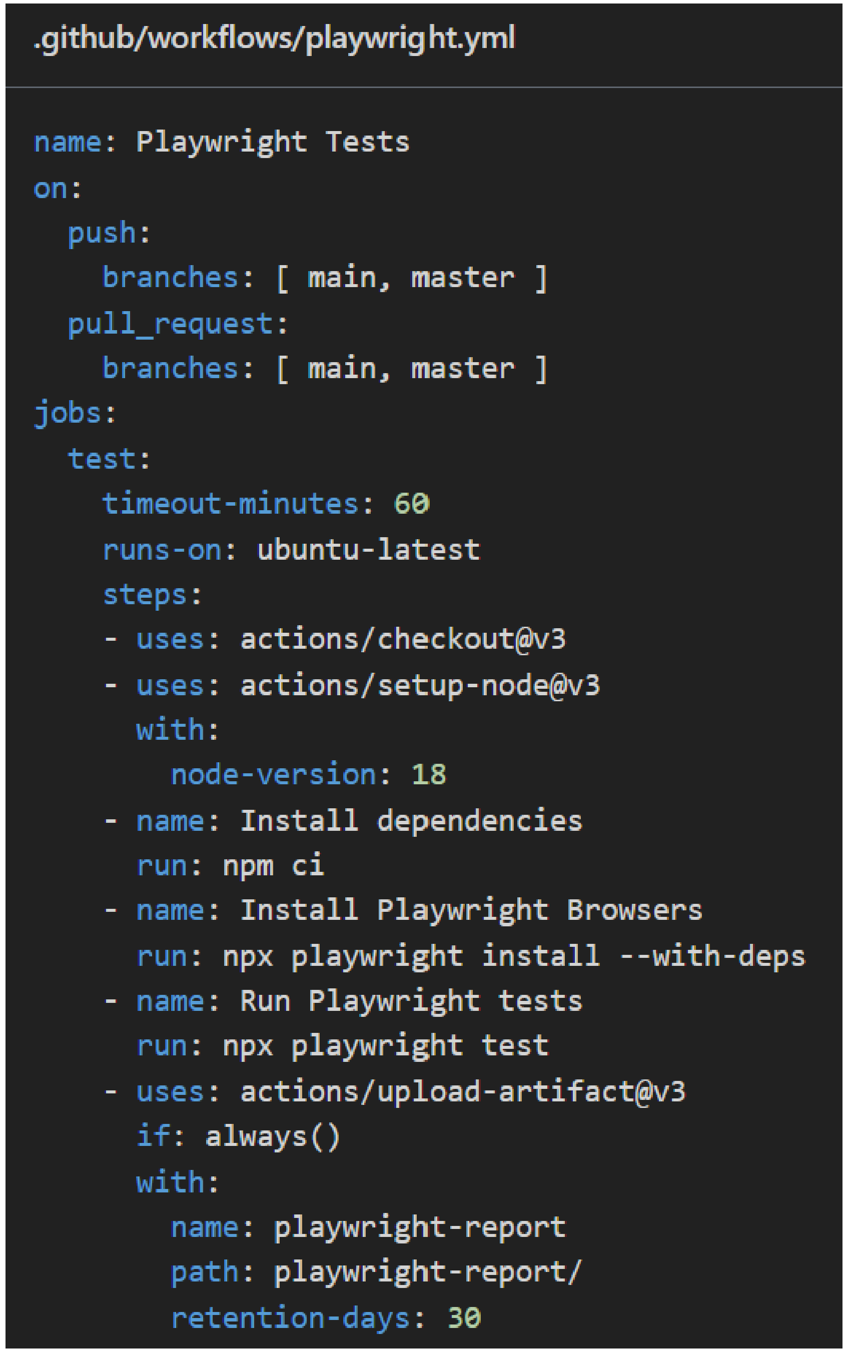
```
npx playwright test
```

CI GitHub Actions

When installing Playwright using the VS Code extension or with npm init playwright@latest you are given the option to add a GitHub Actions.

This creates a playwright.yml file inside a .github/workflows folder containing everything you need so that your tests run on each push and pull request into the main/master branch.

The workflow will install all dependencies, install Playwright and then run the tests. It will also create the HTML report.



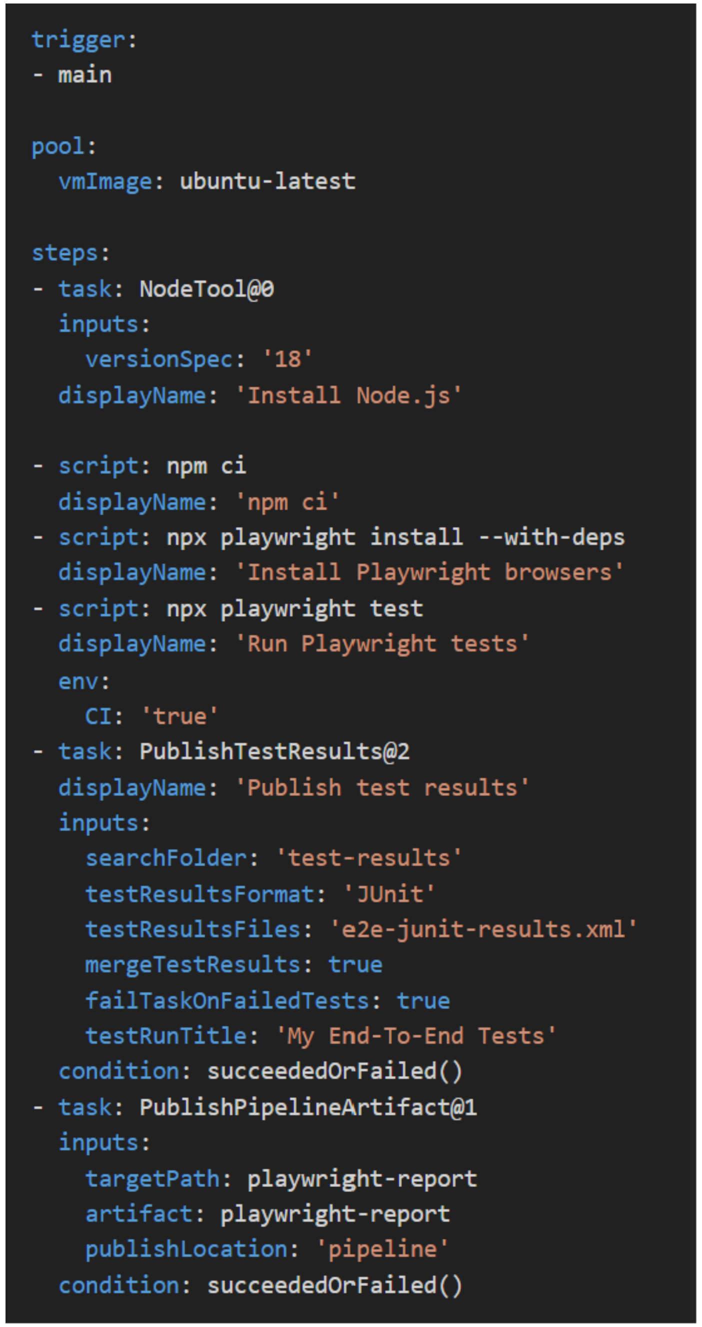
.github/workflows/playwright.yml

```yaml
name: Playwright Tests
on:
  push:
    branches: [ main, master ]
  pull_request:
    branches: [ main, master ]
jobs:
  test:
    timeout-minutes: 60
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3
    - uses: actions/setup-node@v3
      with:
        node-version: 18
    - name: Install dependencies
      run: npm ci
    - name: Install Playwright Browsers
      run: npx playwright install --with-deps
    - name: Run Playwright tests
      run: npx playwright test
    - uses: actions/upload-artifact@v3
      if: always()
      with:
        name: playwright-report
        path: playwright-report/
        retention-days: 30
```

CI Azure Pipelines

The pipeline will install all dependencies, install Playwright and then run the tests. It will also create the HTML report and publish it as an artifacts using ‘PublishTestResults’ task.

The pipeline run fails even if one of the playwright tests fails by adding condition: succeededOrFailed()



```yaml
trigger:
- main

pool:
  vmImage: ubuntu-latest

steps:
- task: NodeTool@0
  inputs:
    versionSpec: '18'
  displayName: 'Install Node.js'

- script: npm ci
  displayName: 'npm ci'
- script: npx playwright install --with-deps
  displayName: 'Install Playwright browsers'
- script: npx playwright test
  displayName: 'Run Playwright tests'
  env:
    CI: 'true'
- task: PublishTestResults@2
  displayName: 'Publish test results'
  inputs:
    searchFolder: 'test-results'
    testResultsFormat: 'JUnit'
    testResultsFiles: 'e2e-junit-results.xml'
    mergeTestResults: true
    failTaskOnFailedTests: true
    testRunTitle: 'My End-To-End Tests'
  condition: succeededOrFailed()
- task: PublishPipelineArtifact@1
  inputs:
    targetPath: playwright-report
    artifact: playwright-report
    publishLocation: 'pipeline'
  condition: succeededOrFailed()
```

Demo

CI Playwright Tests

Questions?

Microsoft