**Publishing Test Results Using Core Actions**

Publishing test results in GitHub Actions is a critical step in continuous integration workflows.

* **Refer Here** for [core actions documentation](https://docs.github.com/en/actions/using-github-hosted-runners/about-github-hosted-runners)
* **Caching** could significantly reduce build times. [Learn more](https://docs.github.com/en/actions/using-workflows/caching-dependencies-to-speed-up-workflows)

**Contexts in GitHub Actions**

GitHub Actions **contexts** are special variables that provide information about the workflow run, job, steps, runner environment, and more.

* **Reference**: [GitHub Contexts Documentation](https://docs.github.com/en/actions/learn-github-actions/contexts)
* **Syntax**: ${{ context.property }}

Common contexts include:

* github
* env
* job
* steps

**Environmental Variables**

You can define environmental variables at different levels:

* **Workflow level**
* **Job level**
* **Step level**

**Reference**: [GitHub Actions Environment Variables](https://docs.github.com/en/actions/learn-github-actions/environment-variables)

Example:

name: ci-workflow

on:

push:

branches:

- main

env:

MY\_WORKFLOW\_VAR: "hello"

jobs:

build:

name: Build Java Code Using Maven

env:

MY\_JOB\_VAR: "job-hello"

runs-on: ubuntu-24.04

steps:

- name: Get the code

uses: actions/checkout@v4

- run: printenv

env:

MY\_STEP\_VAR: "step-hello"

- run: "echo $MY\_WORKFLOW\_VAR, $MY\_JOB\_VAR, $MY\_STEP\_VAR"

env:

MY\_STEP\_VAR: "step-hello"

- name: Set environmental variable

run: "MY\_DYNAMIC\_VAR=Hello"

- name: Setup Java 17

uses: actions/setup-java@v4.5.0

with:

java-version: 17

distribution: temurin

**Note**: Use environment variables with the syntax $VAR\_NAME.

**Secrets and Configuration Values**

* Store secrets securely using GitHub’s Settings > Secrets in your repository.
* **Reference**: [GitHub Secrets Documentation](https://docs.github.com/en/actions/security-guides/encrypted-secrets)

**VS Code Extensions and Debugging**

* GitHub Actions extensions are available in VS Code for managing and running workflows.
* **Reference for Debug Logs**: [Enabling debug logging](https://docs.github.com/en/actions/monitoring-and-troubleshooting-workflows/enabling-debug-logging)
* **Run workflows locally**: Use [Act](https://github.com/nektos/act) for local testing.

**Building Custom GitHub Actions**

You can build your own actions using Docker or JavaScript.

**Creating a Docker Image Action**

* **Repository**: [GitHub Docs - Docker Actions](https://docs.github.com/en/actions/creating-actions/creating-a-docker-container-action)
* **Secrets**: Store image registry credentials securely in GitHub Secrets.
* **Workflow Example with Secret Scanning**: [Secret Scanning Documentation](https://docs.github.com/en/code-security/secret-scanning/about-secret-scanning)

**Exercise:**

* Show failure for any critical issues in the build.
* Push the Docker image to ACR (Azure), ECR (AWS), or GCR (Google).