Controlling Job Execution in GitHub Actions

Dependencies, Conditions, Outputs, Concurrency, Containers, and Matrix Strategies

What you'll learn

Make jobs run serially with `needs`, guard runs with job/step `if` conditions, pass strings between jobs with job `outputs`, throttle work with `concurrency` groups, run inside containers via `container`, and fan out builds with `strategy.matrix`.

Quick Reference

- **Serializing jobs:** use `needs: [job-a, job-b]`.
- **Conditional execution:** job/step `if: <expression>` (e.g., `success()`, `failure()`, `always()`, `cancelled()`).
- **Share small strings across jobs:** write to `GITHUB_OUTPUT` in a step, map that step output to `jobs.<job>.outputs`, then consume via `${{ needs.<job>.outputs.<name> }}`.
- **Concurrency:** `concurrency:{group:..., cancel-in-progress: true|false}` to ensure one run per group.
- **Containers:** `container: <image>` at job-level (plus `credentials`, `env`, `ports`, `volumes`).
- **Matrix:** `strategy.matrix` with `include`/`exclude`; control with `fail-fast` and `max-parallel`.

1 Drop-in Example: Outputs + Needs + Concurrency + Container

.github/workflows/jobs.yml

```yaml
name: Jobs: outputs • needs • concurrency • container
on:
  push:
    branches: [ main ]

jobs:
  build:
    name: Build in Bun container (produce output)
    runs-on: ubuntu-latest
    # Job-level container pulls from Docker Hub by default
    container:
      image: oven/bun:1
      # Optionally authenticate to a private registry:
      # credentials:
      #   username: ${{ secrets.REGISTRY_USER }}
      #   password: ${{ secrets.REGISTRY_PASS }}
      env:
        MY_ENV_VAR: "hello"
      ports:
        - 8080
      volumes:
        - my_volume:/data
    outputs:
      bun_version: ${{ steps.capture.outputs.bun_version }}
    steps:
      - name: Capture Bun version
        id: capture
        run: |
          ver="$(bun --version)"
          echo "bun_version=${ver}" >> "$GITHUB_OUTPUT"

  deploy:
    name: Deploy (needs build output)
    runs-on: ubuntu-latest
    needs: [build]
    # Example condition: only run if not on the main branch
    if: ${{ github.ref != 'refs/heads/main' || always() }}
    steps:
      - name: Show value passed from build
        run: |
          echo "Bun version from build: ${{ needs.build.outputs.bun_version }}"

  concurrency-job-1:
    name: Concurrency A (group=test-concurrency)
    runs-on: ubuntu-latest
    concurrency:
      group: test-concurrency
      cancel-in-progress: false
    steps:
      - run: sleep 15

  concurrency-job-2:
    name: Concurrency B (group=test-concurrency)
    runs-on: ubuntu-latest
    concurrency:
      group: test-concurrency
      cancel-in-progress: false
    steps:
      - run: sleep 15
```

Why this works. `deploy` waits for `build` (`needs`). The `build` job writes a step output to `GITHUB_OUTPUT`, maps it to a job output, and `deploy` consumes it. The two `concurrency-*` jobs share a group so only one runs at a time; the other queues until the first finishes.

Outputs vs. Artifacts

Use **job outputs** for small strings. For files/binaries/reports, use artifacts:

```yaml
- name: Upload build report
  uses: actions/upload-artifact@v4
  with:
    name: report
    path: build/report.txt
```

2 Drop-in Example: Matrix Across OS and Versions

.github/workflows/matrix.yml

```yaml
name: Matrix demo
on:
  push:
    branches: [ main ]

jobs:
  test:
    name: ${{ matrix.runner }} • py ${{ matrix.python-version }}
    runs-on: ${{ matrix.runner }}
    strategy:
      fail-fast: false
      max-parallel: 3
      matrix:
        runner: [ubuntu-latest, windows-latest]
        python-version: ["3.11", "3.12", "3.13"]
        include:
          - runner: macos-latest
            python-version: "3.12"
        exclude:
          - runner: windows-latest
            python-version: "3.13"
    steps:
      - name: Show current combo
        run: echo "Runner=${{ matrix.runner }}, Python=${{ matrix.python-version }}"
```

Expected fan-out. Ubuntu: 3 jobs (3.11, 3.12, 3.13). Windows: 2 jobs (3.11, 3.12) because 3.13 excluded. macOS: 1 job (3.12). Total = 6.

3 Recipes & Patterns (Copy/Paste)

Guard Jobs/Steps with Conditions

```
# Job runs even if dependencies fail:
if: ${{ always() }}

# Only on tags:
if: ${{ startsWith(github.ref, 'refs/tags/') }}

# Skip if commit message contains [skip ci]:
if: ${{ !contains(github.event.head_commit.message, '[skip ci]') }}

# Only on pull requests from forks:
if: ${{ github.event_name == 'pull_request' && github.event.pull_request.head.repo.fork }}
```

Chain Jobs with Multiple Dependencies

```
jobs:
  a: { runs-on: ubuntu-latest, steps: [ { run: echo A } ] }
  b:
    needs: [a]
    runs-on: ubuntu-latest
    steps: [ { run: echo B needs A } ]
  c:
    needs: [a, b]
    runs-on: ubuntu-latest
    steps: [ { run: echo C after A and B } ]
```

Concurrency: Key by Branch or Workflow

```
concurrency:
  group: ${{ github.workflow }}-${{ github.ref }}
  cancel-in-progress: true
```

Containers: Private Registry + Env/Ports/Volumes

```
jobs:
  in-container:
    runs-on: ubuntu-latest
    container:
      image: ghcr.io/acme/my-runtime:1.2.3
      credentials:
        username: ${{ secrets.GHCR_USER }}
        password: ${{ secrets.GHCR_TOKEN }}
      env:
        APP_MODE: ci
      ports: [8080]
      volumes:
        - cache:/cache
    steps:
      - run: my-runtime --version
```

Dynamic Matrix (from Script Output)

```
jobs:
  gen-matrix:
    runs-on: ubuntu-latest
    outputs:
      combo: ${{ steps.mk.outputs.combo }}
    steps:
      - id: mk
        run: |
          # Produce JSON: {"include":[{"runner":"ubuntu-latest","py":"3.12"}]}
          echo 'combo={"include":[{"runner":"ubuntu-latest","py":"3.12"},{"runner":"windows-latest
          ↪   ","py":"3.11"}]}' >>
          ↪   "$GITHUB_OUTPUT"

  use-matrix:
    needs: gen-matrix
    runs-on: ${{ matrix.runner }}
    strategy:
      matrix: ${{ fromJson(needs.gen-matrix.outputs.combo) }}
    steps:
      - run: echo "Runner=${{ matrix.runner }}, py=${{ matrix.py }}"
```

4 Common Pitfalls & Fixes

- **Forgot `id` on a step** before referencing `steps.<id>.outputs.<name>`.
- **Not writing to `GITHUB_OUTPUT`**: use `echo "k=v" » $GITHUB_OUTPUT`.
- **Using outputs for files**: prefer artifacts for files/binaries.
- **Matrix blow-up**: add `exclude` and tune `max-parallel`.
- **Container networking confusion**: job-level `ports` expose container ports on the runner; services need `services:`.
- **Long queues on hot branches**: set `cancel-in-progress: true` with a branch-keyed group.

5 Debugging Tips

```
- name: Dump context (useful for authoring)
  if: ${{ runner.os == 'Linux' }}
  run: |
    echo "Ref=${GITHUB_REF}"
    echo "SHA=${GITHUB_SHA}"
    echo "Actor=${GITHUB_ACTOR}"
    echo "Workflow=${GITHUB_WORKFLOW}"
```