Reusable Workflows in GitHub Actions

Reusable workflows in GitHub Actions allow you to create modular, shareable workflow components that can be utilized across multiple repositories and workflows. These reusable workflows can help streamline your CI/CD processes and make your workflows more maintainable. This README provides an overview of key concepts and usage.

Creating a Reusable Workflow

1. **Workflow File:** Reusable workflows are defined in YAML-formatted files, similar to regular workflow files. Place these files in the `.github/workflows` directory of your repository.

2. `on` **Event:** For a workflow to be reusable, include `workflow_call` in its `on` event. This specifies that the workflow can be called by other workflows.

3. **Inputs and Secrets:** Reusable workflows can accept inputs and secrets, which can be passed from the caller workflow and used within the called workflow. Define these inputs and secrets using the `inputs` and `secrets` keywords.

Here's an example:

```
on:
  workflow_call:
    inputs:
      config-path:
        required: true
        type: string
    secrets:
      token:
        required: true
```

Calling a Reusable Workflow

To call a reusable workflow within a workflow, use the uses keyword. Unlike actions, reusable workflows are called directly within a job, not from within job steps:

```
jobs:
  call-reusable-workflow:
    uses: ./.github/workflows/reusable-workflow.yaml
```

Passing Inputs and Secrets

To pass inputs to a reusable workflow, use the `with` keyword, and to specify secrets use the `secrets` keyword:

```
jobs:
  call-reusable-workflow:
    uses: ./.github/workflows/reusable-workflow.yaml
    with:
      input-value: "hello world"
    secrets:
      secret-value: "I will not be visible"
```

Using Outputs from a Reusable Workflow

You can capture outputs from a reusable workflow and use them in the caller workflow. The process is very similar to using outputs between different dependent jobs in the same workflow file, but here you also have to specify the outputs under the `outputs` keyword in the `workflow_call` definition.

```
# Reusable Workflow
on:
  workflow_call:
    outputs:
      my-output:
        value: ${{ jobs.my-job.outputs.my-output }}

jobs:
  my-job:
    runs-on: ubuntu-latest
    outputs:
      my-output: ${{ steps.my-step.outputs.my-output }}
    steps:
      - name: My Step
        id: my-step
        run: |
          # Do something
          echo "my-output=my-value" >> "$GITHUB_OUTPUT"
```

```
# Caller Workflow
jobs:
  call-reusable-workflow:
    uses: ./.github/workflows/reusable-workflow.yaml
  use-outputs:
    needs: call-reusable-workflow
    runs-on: ubuntu-latest
    steps:
      - run: echo ${{ needs.call-reusable-workflow.outputs.my-output }}
```

Testing Reusable Workflow

1. Reusable Workflow Definition (23-01-Reusable-Workflows):

```yaml
name: 23-01-Reusable-Workflows - Reusable Definition

on:
  workflow_call:
    inputs:
      target-directory:
        type: string
        required: true
    outputs:
      build-status:
        description: The status of the build process
        value: ${{ jobs.deploy.outputs.build-status }}
      url:
        description: The url of the deployed version
        value: ${{ jobs.deploy.outputs.url }}

jobs:
  deploy:
    runs-on: ubuntu-latest
    outputs:
      build-status: ${{ steps.build.outputs.build-status }}
      url: ${{ steps.deploy.outputs.url }}
    steps:
      - name: Checkout repo
        uses: actions/checkout@v4
      - name: Build
        id: build
        run: |
          echo "Building using directory ${{ inputs.target-directory }}"
          echo "build-status=success" >> "$GITHUB_OUTPUT"
      - name: Deploy
        id: deploy
        run: |
          echo "Deploying build artifacts"
          echo "url=https://www.google.com" >> "$GITHUB_OUTPUT"
```

- This workflow defines a reusable job named `deploy`.
- It accepts an input parameter `target-directory` specifying the target directory for deployment.
- The `deploy` job performs build and deployment tasks.
- It outputs `build-status` and `url` for use in other workflows.

2. Workflow Triggering the Reusable Workflow (23-02-Reusable-Workflows):

```yaml
name: 23-02-Reusable Workflows
```

```
on:
  workflow_dispatch:

jobs:
  deploy:
    uses: ./.github/workflows/21-01-reusable-workflows.yaml
    with:
      target-directory: dummmy-dir
  print-outputs:
    runs-on: ubuntu-latest
    needs: deploy
    steps:
      - name: Print outputs
        run: |
          echo "Build status: ${{ needs.deploy.outputs.build-status }}"
          echo "URL: ${{ needs.deploy.outputs.url }}"
```

- This workflow triggers the reusable `deploy` job defined in `23-01-Reusable-Workflows`.
- It passes the `target-directory` input to the `deploy` job.
- After the `deploy` job completes, it prints the outputs.

3. Workflow Triggering the Reusable Workflow and External Workflow (23-03-Reusable-Workflows):

```
name: 23-03-Reusable Workflows

on:
  workflow_dispatch:

jobs:
  deploy:
    uses: ./.github/workflows/23-01-reusable-workflows.yaml
    with:
      target-directory: dummmy-dir
  e2e-tests:
    uses: PadmanabhanSaravanan/github-actions-course-example-
e2e/.github/workflows/e2e.yaml@main
    needs: deploy
    secrets:
      access-token: ${{ secrets.GH_TOKEN }}
```

- This workflow also triggers the reusable `deploy` job.
- Additionally, it triggers an external workflow (`e2e-tests`) after the `deploy` job.
- The `e2e-tests` workflow is sourced from an external repository using a specific path and version.
- It provides a secret `access-token` to the `e2e-tests` workflow.

4. Workflow Testing Steps:

- Commit Workflows:

    - Commit the `23-01-Reusable-Workflows.yml` and `23-02-Reusable-Workflows.yml` and `23-03-Reusable-Workflows.yml` files to the `.github/workflows` directory in your repository.

- Trigger Workflows:
    - Manually trigger the `23-02-Reusable-Workflows` workflow from the GitHub Actions UI.
    - Verify that the `deploy` job runs successfully and prints the outputs.
- Verify Outputs:
- Check the logs of the `print-outputs` job in the `23-02-Reusable-Workflows` workflow to ensure that the outputs are printed correctly.
- Trigger External Workflow:
    - Manually trigger the `23-03-Reusable-Workflows` workflow.
    - Ensure that the `deploy` job runs and triggers the `e2e-tests` workflow from the external repository.
- Review Logs:
    - Review the logs of the triggered workflows to verify the execution flow and outputs.