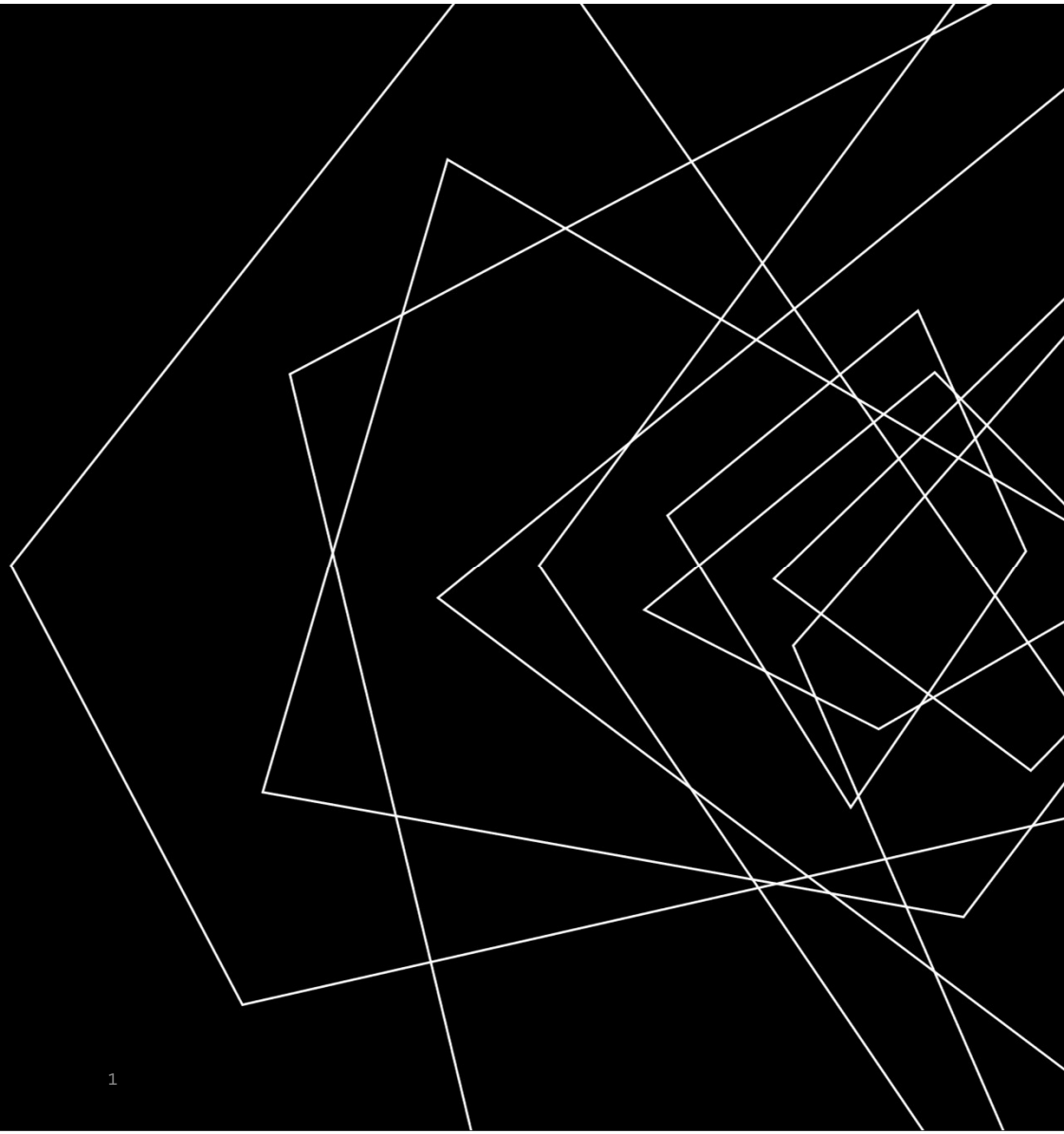
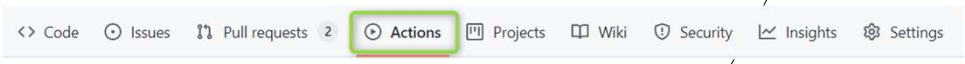
GitHub Actions

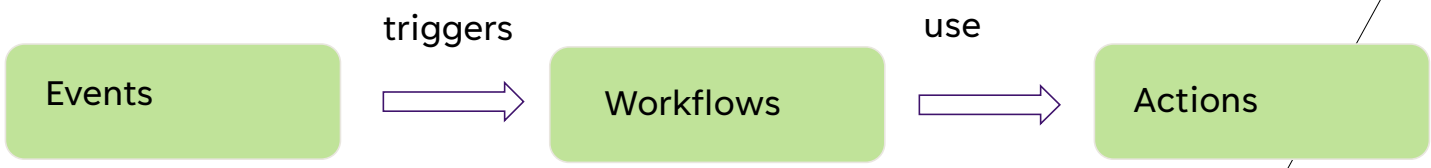


Keerthi V

WHAT IS GITHUB ACTIONS?

1. Actions help you automate tasks within your software development
2. They are event driven.





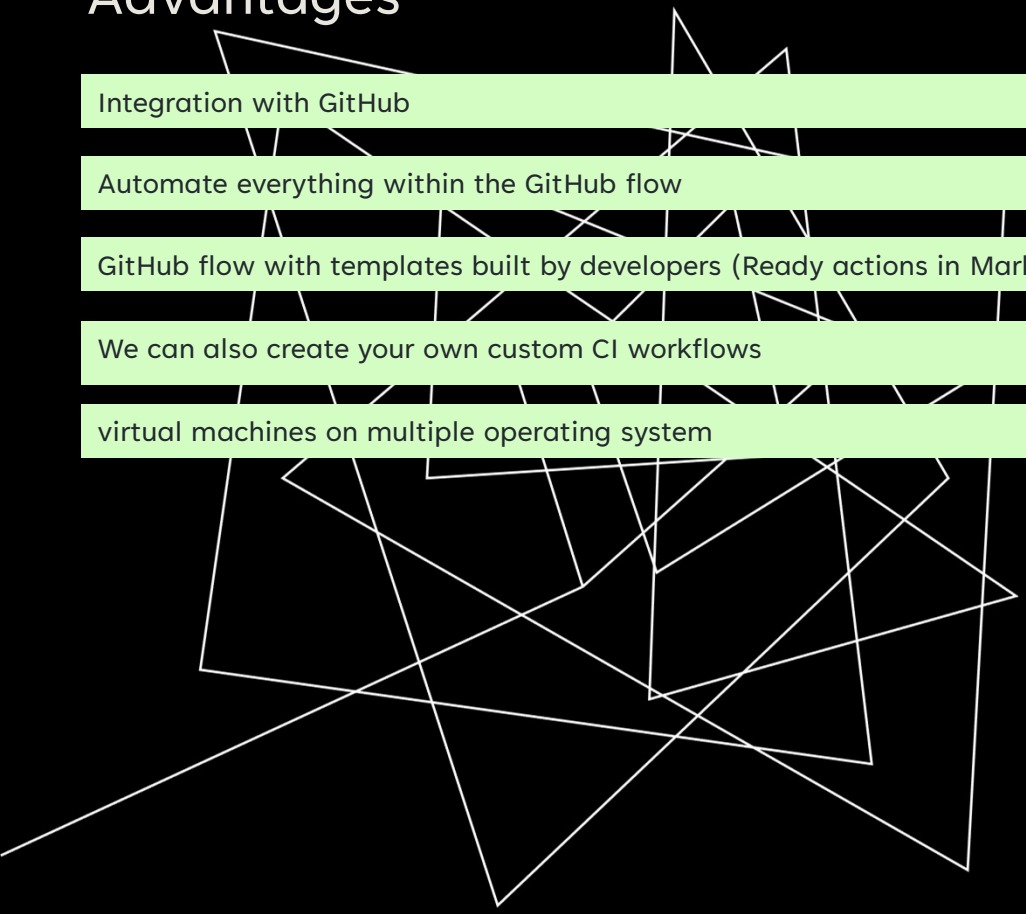
Events trigger workflows, e.g. a push to a branch

PR Merged
PR created
push
Issue created

label
deploy
validate
Assign

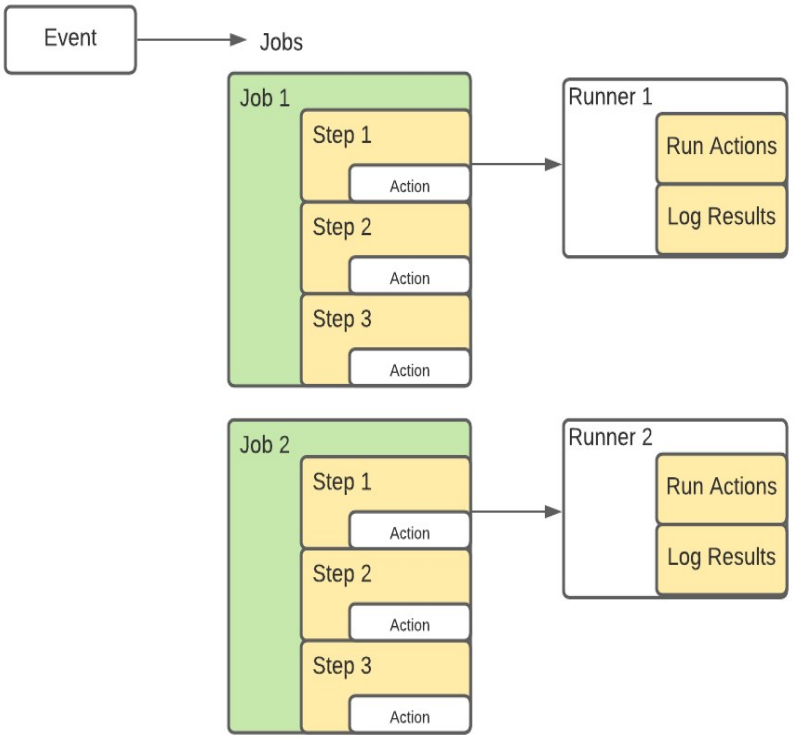
The term "***GitHub Actions***" include all components, not just the Actions themselves

Advantages



- Integration with GitHub
- Automate everything within the GitHub flow
- GitHub flow with templates built by developers (Ready actions in Marketplace)
- We can also create your own custom CI workflows
- virtual machines on multiple operating system

Key Components

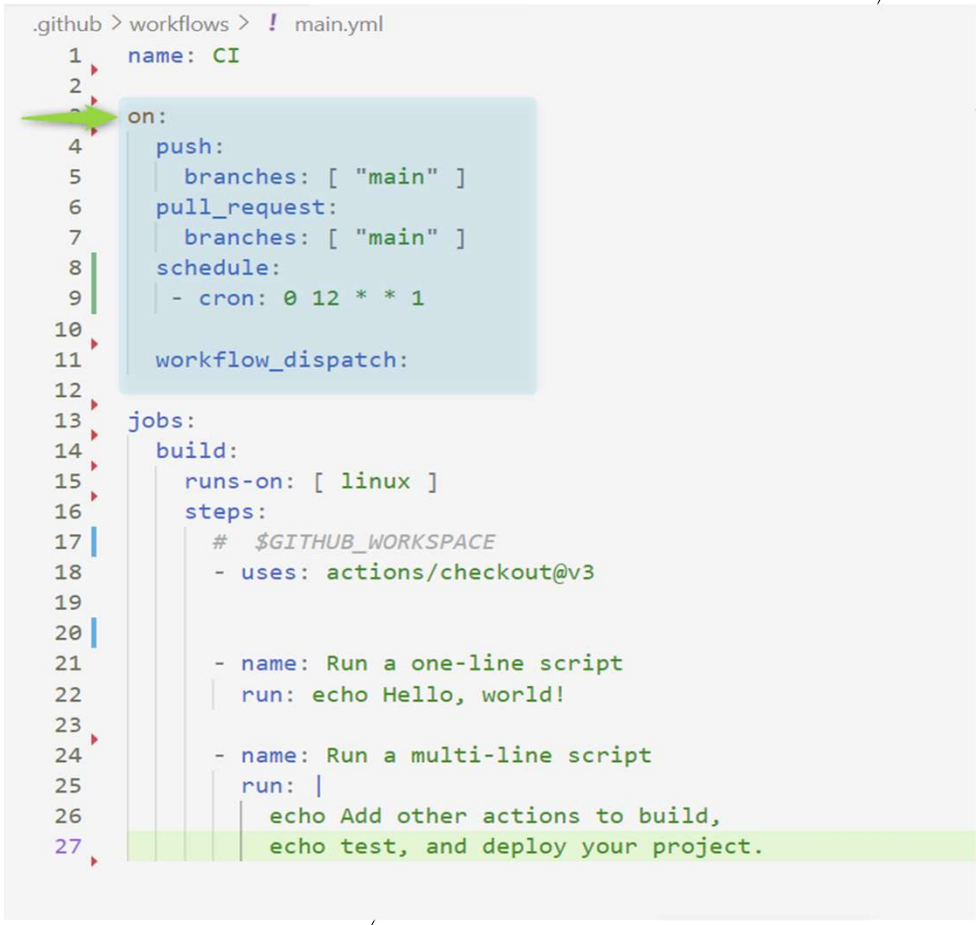


<repo>.github/workflows/main.yml

```yaml
name: CI

# Controls when the workflow will run
on:
  # Triggers the workflow on push or pull request events but only for the "main" branch
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

  # Allows you to run this workflow manually from the Actions tab
  workflow_dispatch:

# A workflow run is made up of one or more jobs that can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: [ linux ]

    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
      - uses: actions/checkout@v3

      # Runs a single command using the runners shell
      - name: Run a one-line script
        run: echo Hello, world!

      # Runs a set of commands using the runners shell
      - name: Run a multi-line script
        run: |
          echo Add other actions to build,
          echo test, and deploy your project.
```

Events

An Event is specific activity that triggers a workflow.



Events

schedule

```
name: Do things every 5 minutes
on:
  schedule:
    - cron: "*/5 * * * *"
```

pull_request

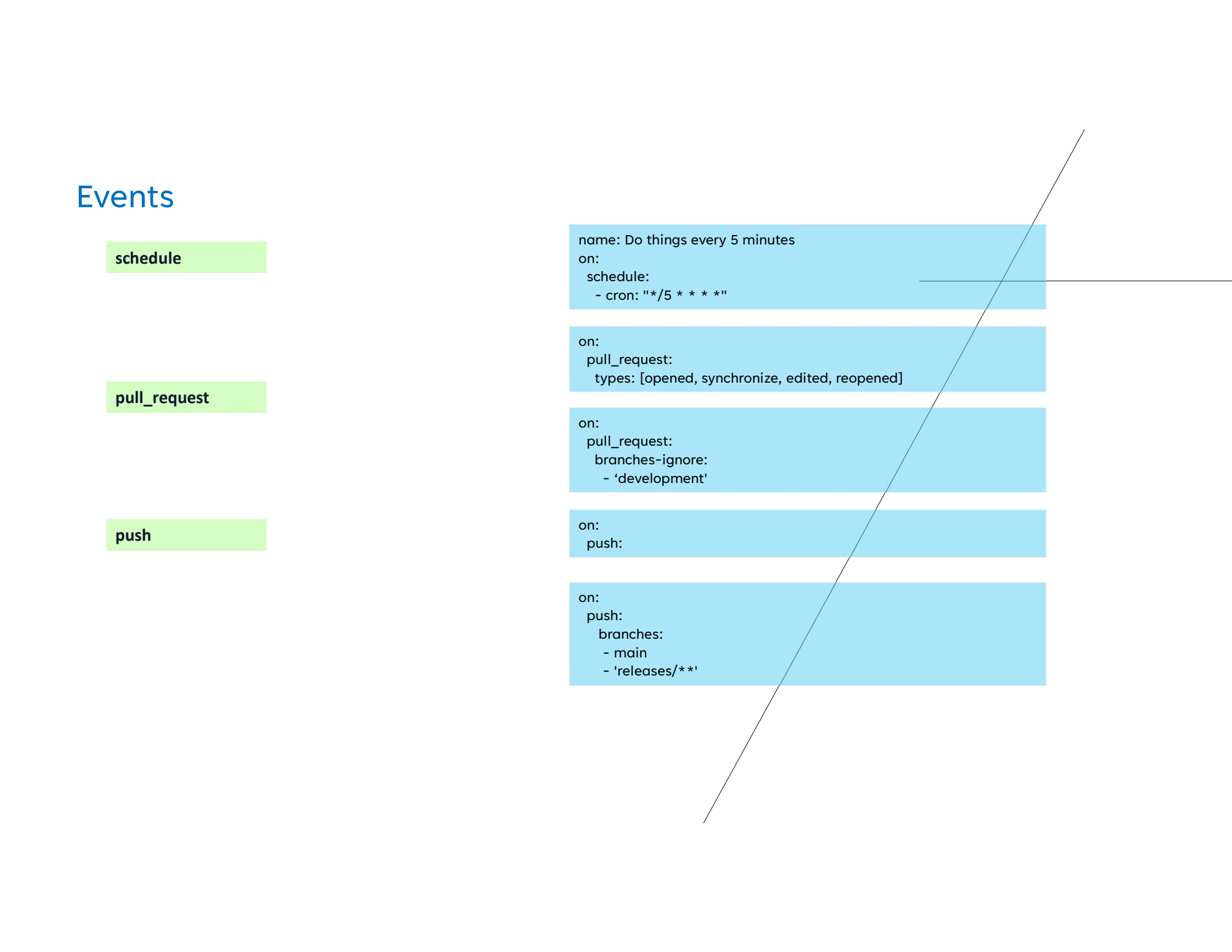
```
on:
  pull_request:
    types: [opened, synchronize, edited, reopened]
```

```
on:
  pull_request:
    branches-ignore:
      - ‘development'
```

push

```
on:
  push:
```

```
on:
  push:
    branches:
     - main
     - 'releases/**'
```



Events

workflow_dispatch

repository_dispatch

workflow_call

```
on:
  workflow_dispatch:
```

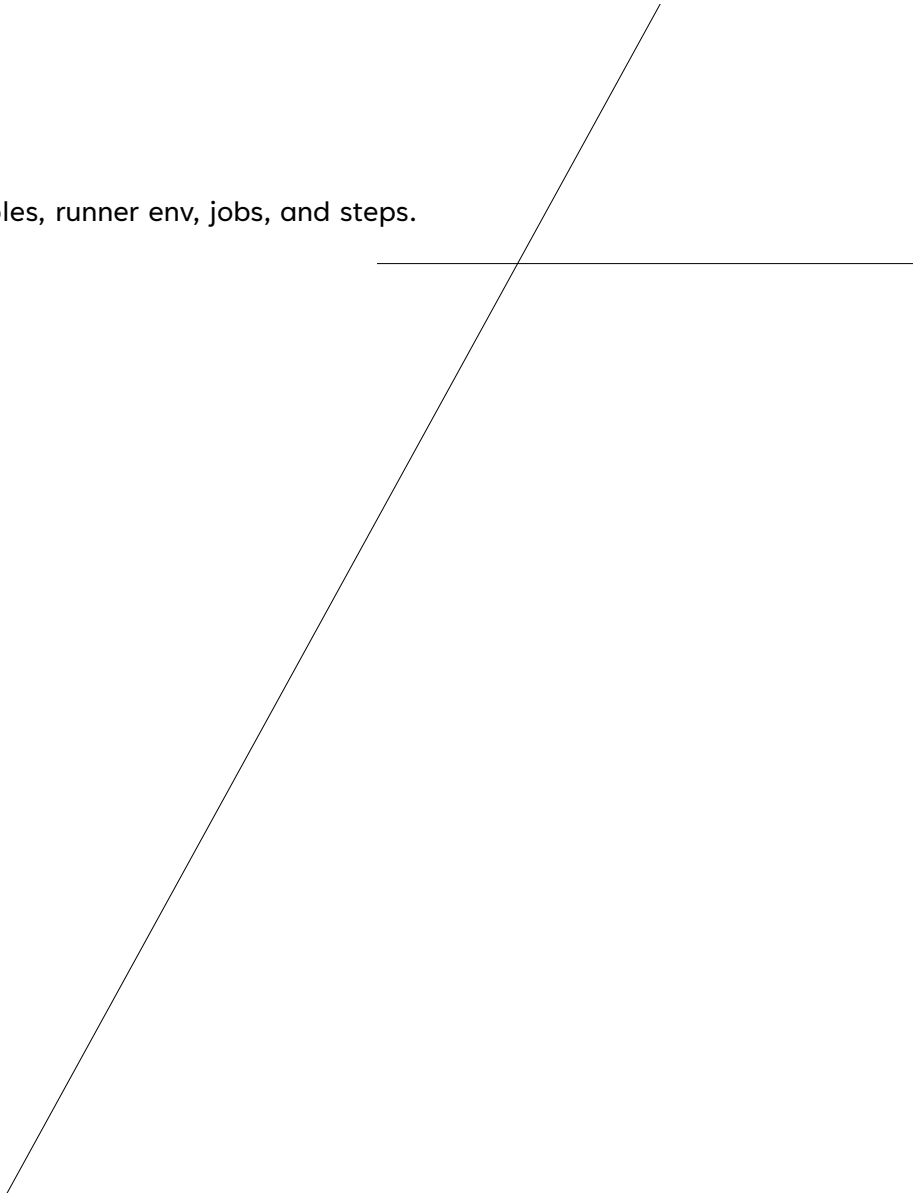
```
on:
  repository_dispatch:
    types: [test_result]
```

```
on: workflow_call
```

Context

Contexts : Contexts are a way to access information about workflow runs, variables, runner env, jobs, and steps.

- GitHub
- Runner
- ENV
- Secrets
- Needs



GitHub Context :

When you run any workflow, Github Actions exposes a set of default environment variables.

```
1  ▶ Run echo "HOME: ${HOME}"
13 HOME: /home/Azurelogin
14 GITHUB_WORKFLOW: Environment variables
15 GITHUB_REPOSITORY: [redacted]/githubactions
16 GITHUB_EVENT_NAME: workflow_dispatch
17 GITHUB_WORKSPACE: /home/Azurelogin/[redacted]/_work/githubactions/githubactions
18 GITHUB_SHA: b33fe249b487e0a51522db02099fbe86cee70783
19 GITHUB_REF: refs/heads/main
20 GITHUB_ACTION: __run
21 GITHUB_ACTIONS: true
```

```yaml
jobs:
  context:
    runs-on: linux
    steps:
      - name: Run normal CI
        run: |
          echo github workspace: ${{ github.workspace }}
          echo github.ref_name: ${{ github.ref_name }}
          echo github.event_name: ${{ github.event_name }}
          echo github.actor: ${{ github.actor }}

      - run: echo 'Hi ${{ env.name }}'  # Hi keerthi
      - run: echo 'Hi ${{ env.name }}'  # Hi kim
        env:
          name: kim
```

ENV

SCOPE :

- The entire **workflow**, by using env at the top level of the workflow file.
- The contents of a **job** within a workflow, by using jobs.<job_id>.env.
- A specific **step** within a job, by using jobs.<job_id>.steps[*].env.

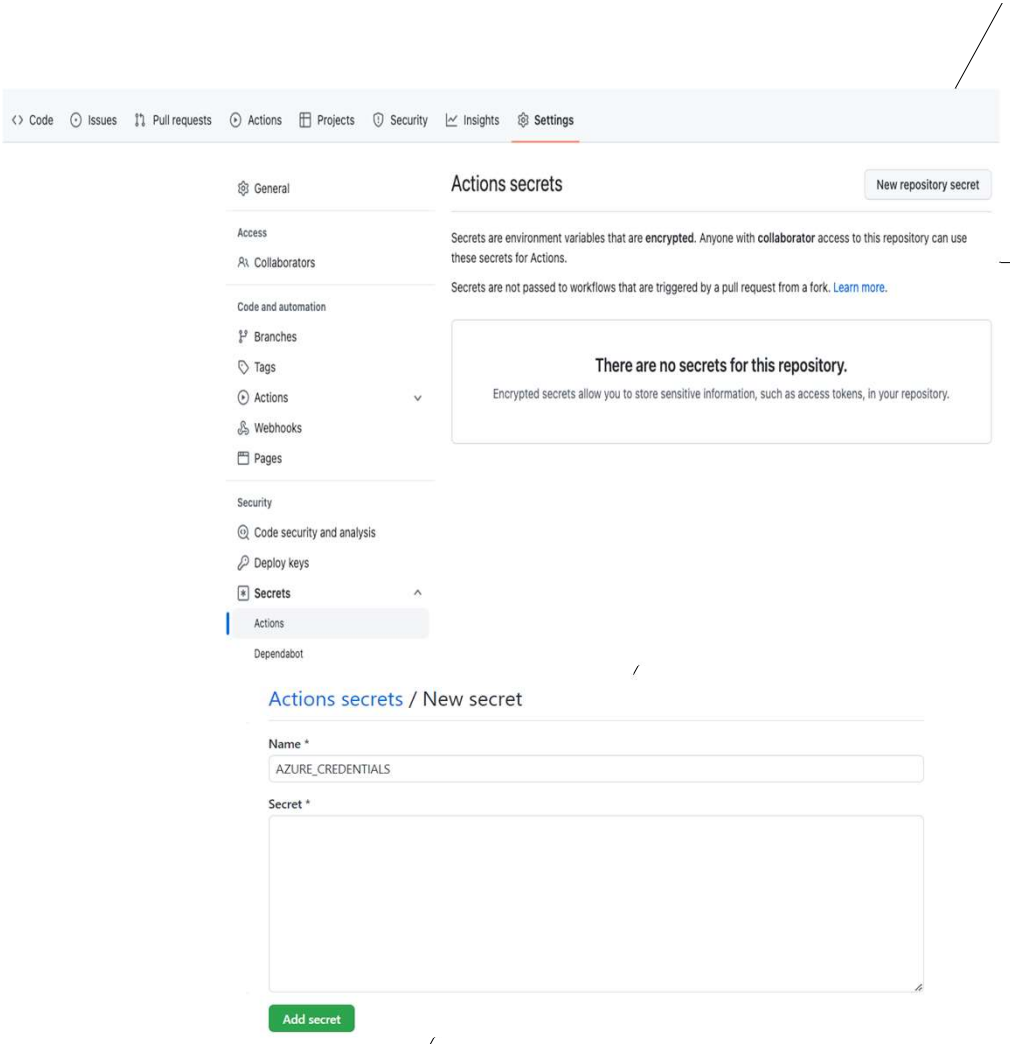
```yaml
on:
  workflow_dispatch:

env:
  name: keerthi

jobs:
  context:
    runs-on: linux
    steps:
      - name: Run normal CI
        run: |
          echo github workspace: ${{ github.workspace }}
          echo github.ref_name: ${{ github.ref_name }}
          echo github.event_name: ${{ github.event_name }}
          echo github.actor: ${{ github.actor }}

      - run: echo 'Hi ${{ env.name }}'  # Hi keerthi
      - run: echo 'Hi ${{ env.name }}'  # Hi kim
        env:
          name: kim

  envcontext:
    runs-on: linux
    env:
      name: ki
    steps:
      - run: echo 'Hi ${{ env.name }}'
```

ki

secrets

GitHub Secrets are encrypted and allow you to store sensitive information, such as access tokens, in your repository.

```
steps:
  - name: Azure details
    with: # Set Azure credentials secret as an input
      credentials: ${{ secrets.AZURE_CREDENTIALS }}
    env: # Or as an environment variable
      credentials: ${{ secrets.AZURE_CREDENTIALS }}
```



Expressions

if

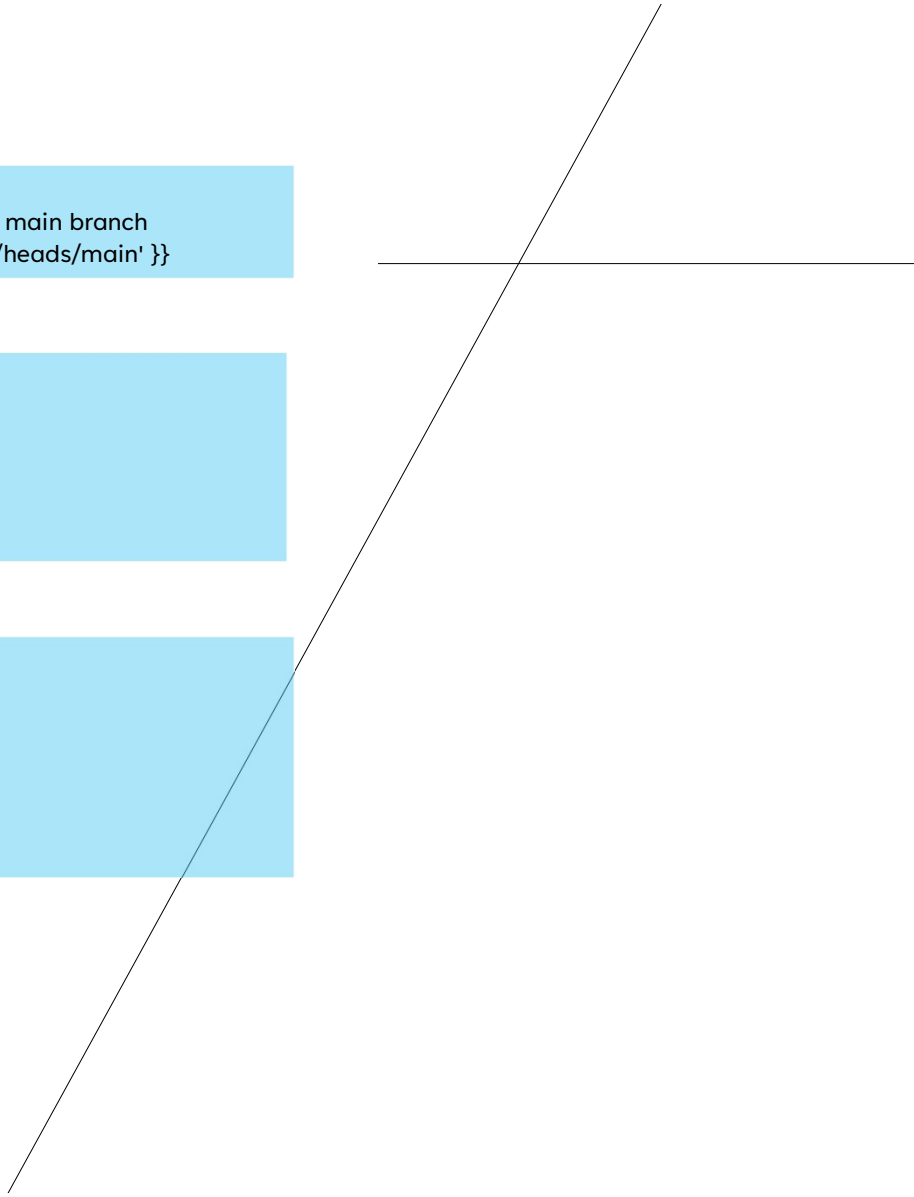
```
steps:
    - name: Configuration for main branch
      if: ${{ github.ref == 'refs/heads/main' }}
```

needs

```
jobs:
  job1:
  job2:
    needs: job1
  job3:
    needs: [job1, job2]
```

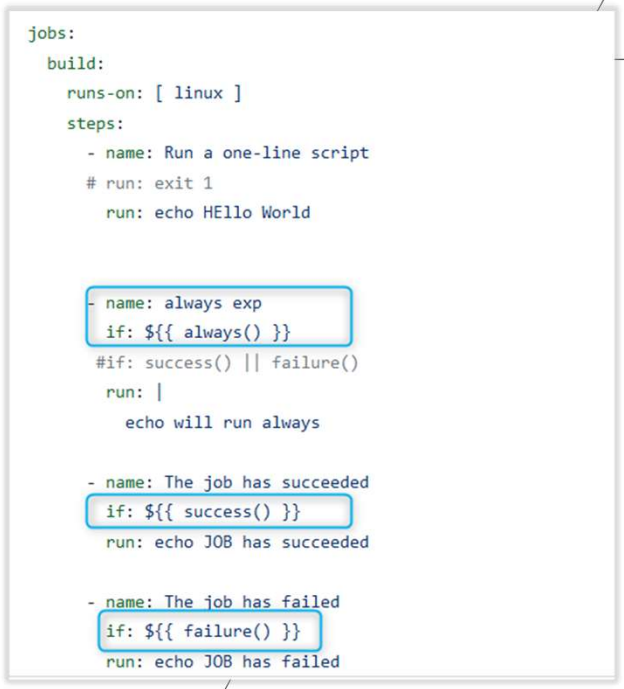
Continue-on-error: true

```
- name: Job fail
  continue-on-error: true
  run |
    exit 1
- name: Next job
  run |
    echo Hello
```

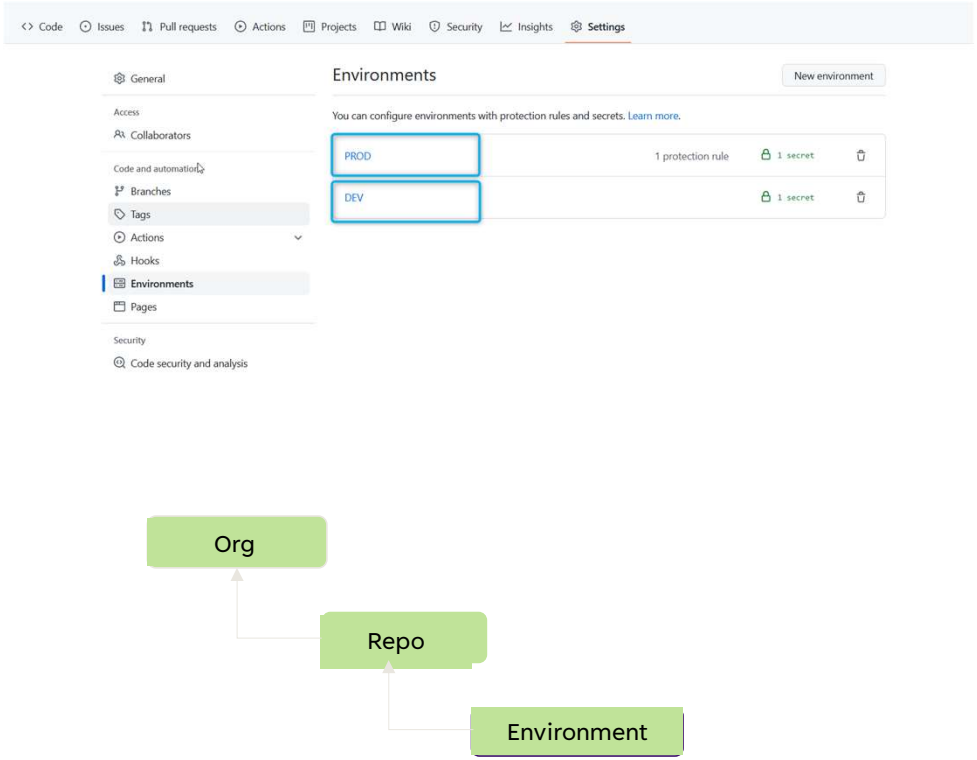


Status check Functions

success()
returns true if no previous steps have failed or have been canceled.
always()
returns true no matter if a step failed or the workflow was cancelled.
failure()
returns true if any previous step failed.



Environments



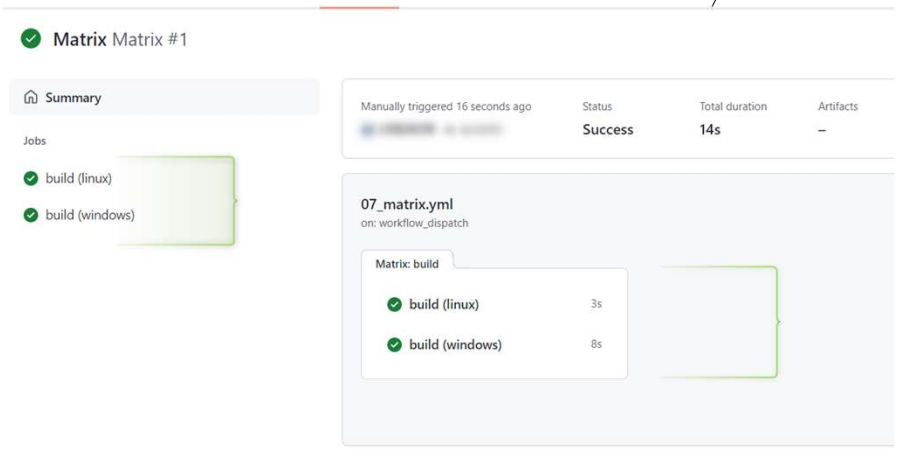

```yaml
name: environment and Protection

on:
  workflow_dispatch:


jobs:
  greet:
    runs-on: [ windows ]
    outputs:
      environment: ${{ steps.check.outputs.BRANCH_ENV }}
    steps:
     - name: Check environment
       id: check
       run: |
        if ($env:GITHUB_REF -ne "refs/heads/main") {
          echo "IS NOT main branch"
          echo "::set-output name=BRANCH_ENV::DEV"
          }
        else {
          echo "IS main branch"
          echo "::set-output name=BRANCH_ENV::PROD"
        }
       shell: powershell
  check:
    runs-on: [ Windows ]
    needs: greet
    environment:  ${{ needs.greet.outputs.environment }}
    steps:
     - name: env names
       run: |
         echo ${{ needs.greet.outputs.environment }}
         echo "$SECRETS_CONTEXT"
       env:
          SECRETS_CONTEXT: ${{ toJson(secrets) }}
```

Matrix

A matrix strategy lets you use variables in a single job definition to automatically create multiple job runs that are based on the combinations of the variables.

For example, you can use a matrix strategy to test your code in multiple versions of a language or on **multiple operating systems**

```yaml
jobs:
  build:
    runs-on: ${{ matrix.os }}
    strategy:
      matrix:
        os: [linux, windows]
```



Runner

GitHub-hosted runners

To use a GitHub-hosted runner, create a job and use runs-on to specify the type of runner that will process the job, such as ubuntu-latest, windows-latest, or macos-latest.

Self-hosted runners

You can host your own runners and customize the environment used to run jobs in your GitHub Actions workflow

- Repository-level runners are dedicated to a single repository. (You must be the repository owner)
- Organization-level runners can process jobs for multiple repositories in an organization. (You must be the organization owner)
- Enterprise-level runners can be assigned to multiple organizations in an enterprise account. (organization owner, or have admin access to the repository.)

Marketplace Actions

- uses: actions/checkout@v3

- uses: actions/javascript-action@v1.0.1

```yaml
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      # This step checks out a copy of your repository.
      - uses: actions/checkout@v3
      # This step references the directory that contains the action.
      - uses: ./.github/actions/hello-world-action
```

https://github.com/marketplace

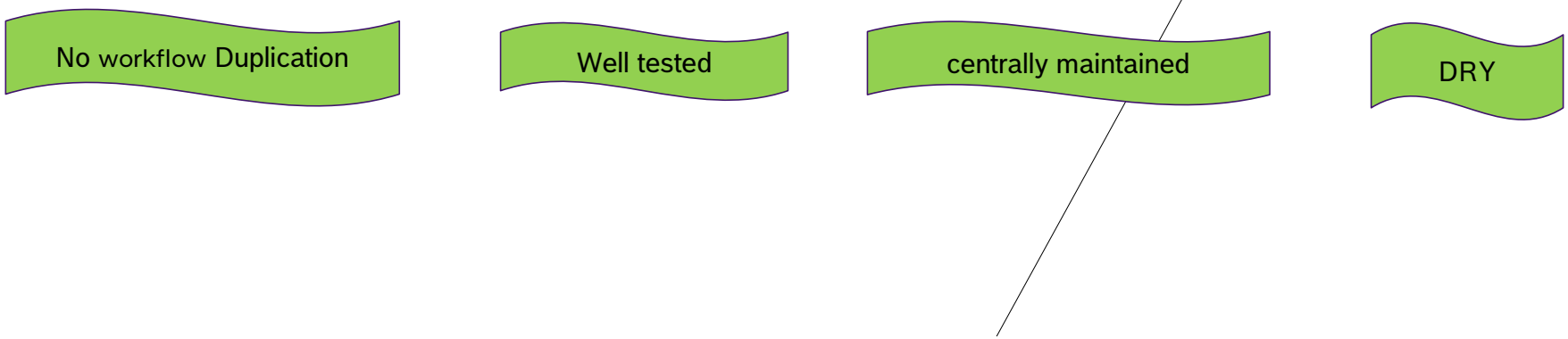
Reusable workflow features

A reusable workflow is similar like any GitHub Actions workflow .
One difference is it includes a “**workflow_call**” trigger.

```
on:
  workflow_call:
```

You can then reference this workflow in another workflow

```
Uses:
 USER_OR_ORG_NAME/REPO_NAME/.github/workflows/REUSABLE_WORKFLOW_FILE.yml@TAG_OR_BRANCH
```



THANK YOU

Workflow :

- Workflows are basically pipelines or automated procedure.
- They are made up of one or more jobs and can be scheduled or trigg by an event
- Workflows are defined in the .github/workflows directory

Jobs :

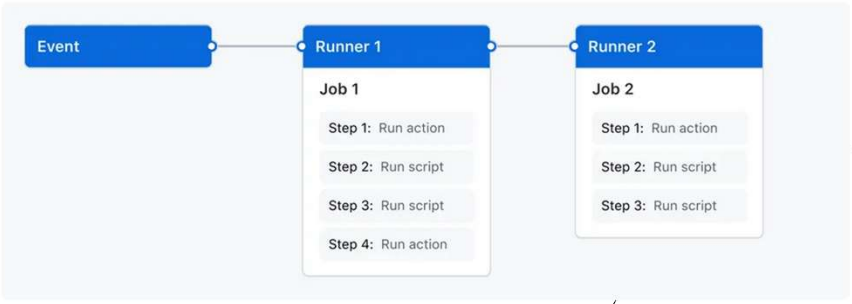
- A job is a set of steps that execute on the same runner.

Steps:

- A step is an individual task that can run commands in a job
- A step can be either an action or a shell command

Runner :

- Runners are processes on a server that run the workflow when it's triggered.
- You can use a runner hosted by GitHub or Self hosted.

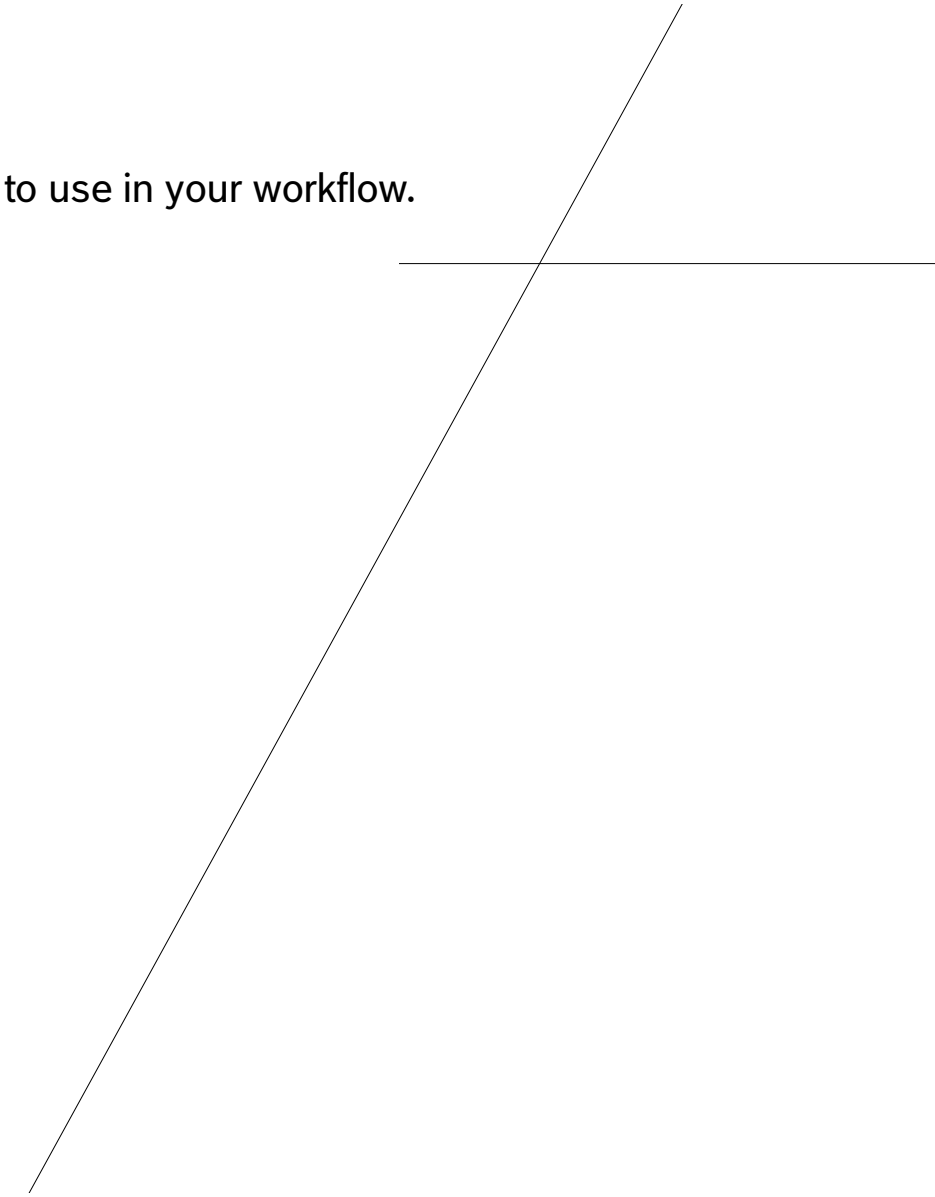


GITHUB_TOKEN secret

GitHub automatically creates a unique **GITHUB_TOKEN** secret to use in your workflow.

Examples :

- include passing the token as an input to an action
- using it to make an authenticated GitHub API request



Matrix

A matrix strategy lets you use variables in a single job definition to automatically create multiple job runs that are based on the combinations of the variables.

For example, you can use a matrix strategy to test your code in multiple versions of a language or on **multiple operating systems**

Using a single-dimension matrix

```yaml
jobs:
  build:
    runs-on: ${{ matrix.os }}
    strategy:
      matrix:
        os: [linux, windows]
```

Using a multi-dimension matrix

```yaml
jobs:
  example_matrix:
    strategy:
      matrix:
        os: [ubuntu-22.04, ubuntu-20.04]
        version: [10, 12, 14]
    runs-on: ${{ matrix.os }}
    steps:
      - uses: actions/setup-node@v3
        with:
          node-version: ${{ matrix.version }}
```