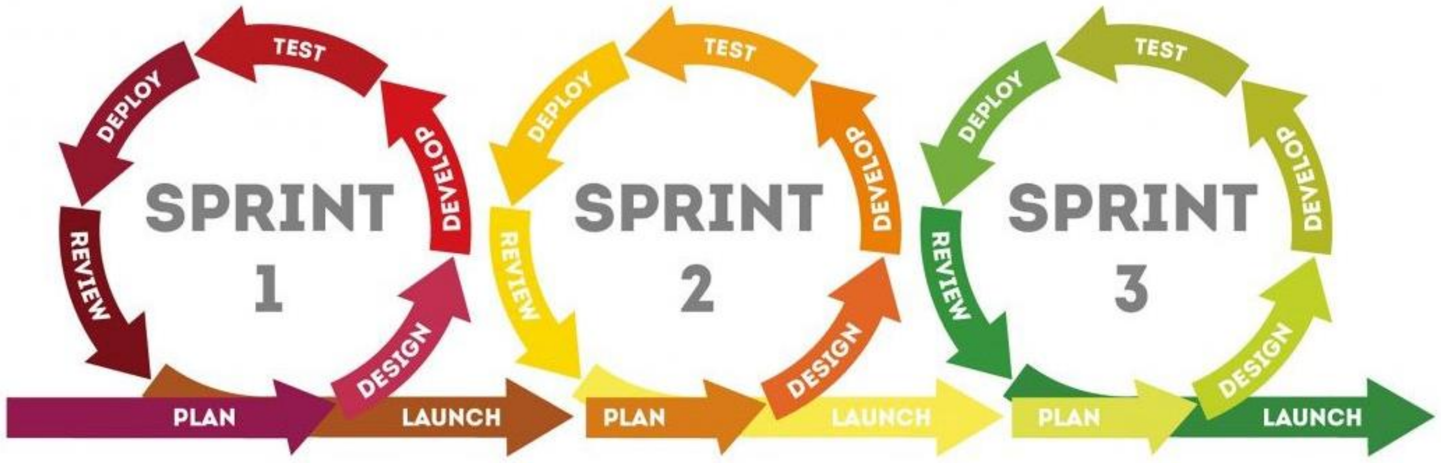
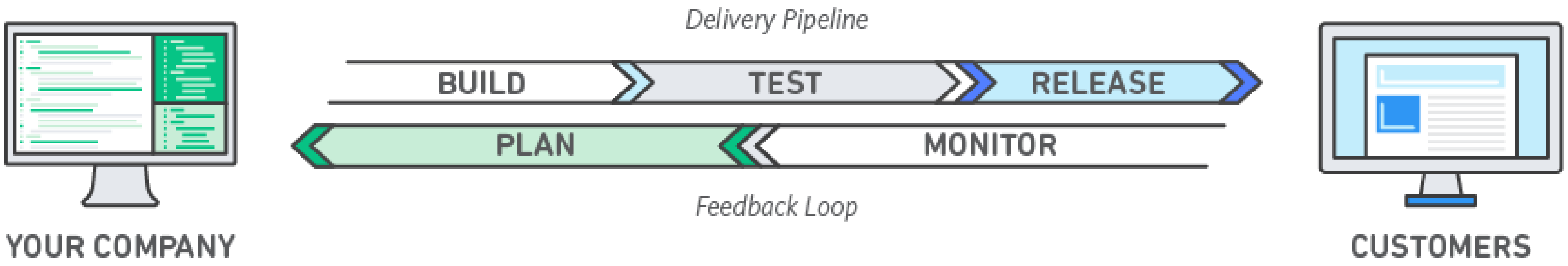
Continuous Integration
Continuous Delivery

What is Agile?



- Agile is a software management framework that focuses on incremental and iterative steps
- Focuses on sprints which are a short-term development cycle
- Goal is to adapt to change by prioritizing manageable and deliverable work

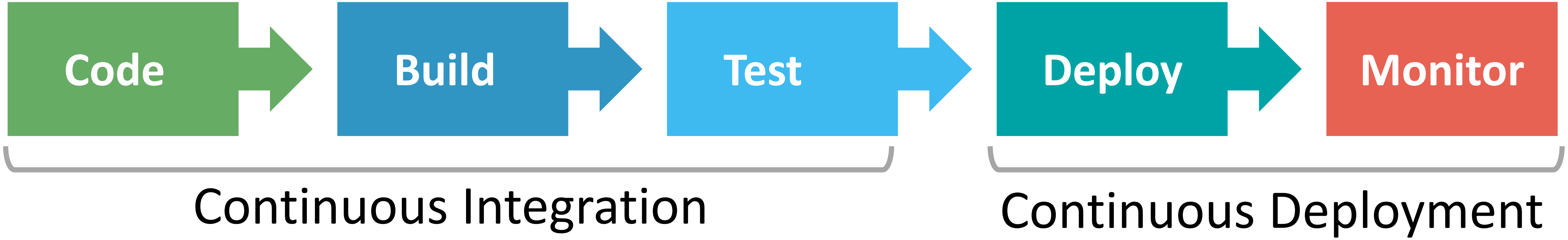
What is DevOps?



- DevOps is a combination of culture, practices and tools
- Aim is to shorten the systems development lifecycle
  - By streamlining software building, testing and release
- Benefit is the improve and evolve applications at a faster pace

Image from https://aws.amazon.com/devops/what-is-devops/

What is CI/CD?



- Refers to the practice of continually building and delivering an application
  - 2 phase process
- Continuous Integration where modules that make up an application are continuously tested and integrated
- Continuous Delivery is the practice of ensuring that the code is always at a deployable state
- Leverage tools to automate the integration and deployment

Relationship Between Agile, DevOps, CICD

Agile

- Focus on Process
- Highlighting Change
- Accelerate Delivery

DevOps

- Focus on Culture
- Highlighting Roles
- Emphasize Responsiveness

CI/CD

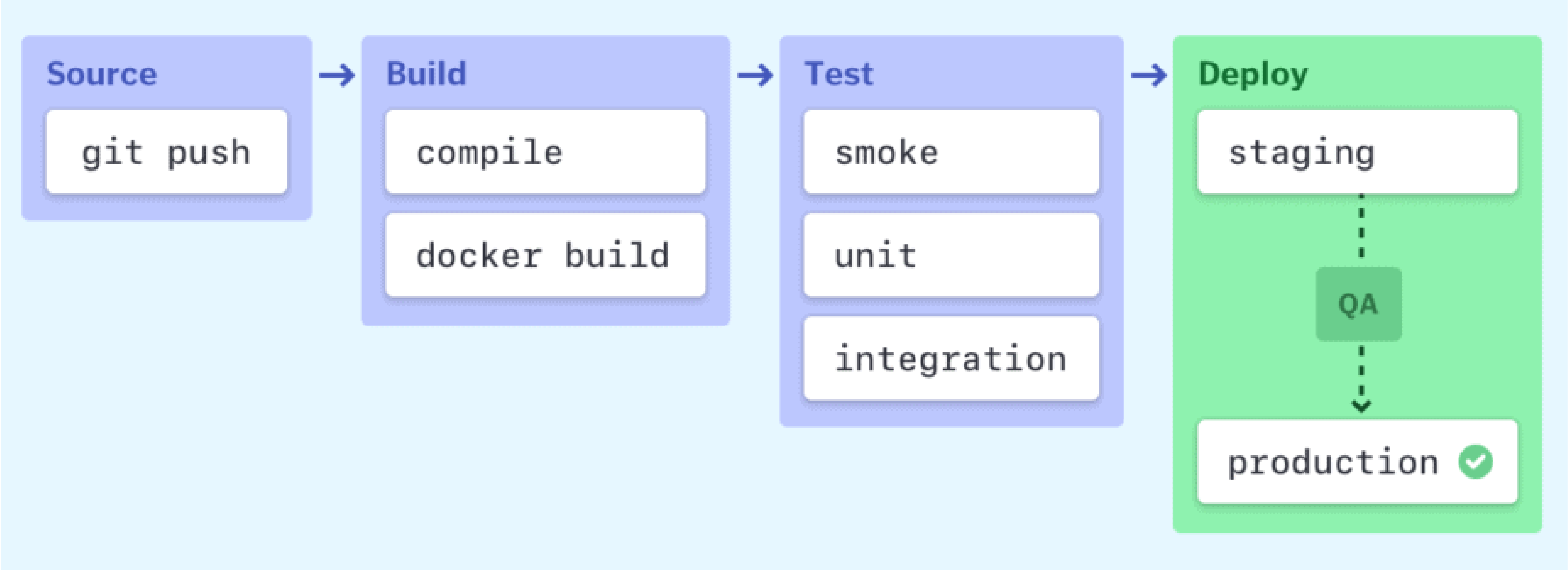
- Focus on Software Defined Lifecycle
- Highlighting Tools
- Adopt Automation

CI/CD

- CI/CD is a strategy used by DevOps practitioners to deliver responsiveness
- CI/CD consist of 2 parts
  - Continuous integration where code are merged when developers pushed their work into the repository
  - Continuous delivery where application delivery is automated once it passes the CI stage
- Automated integration and delivery is triggered by certain event
  - Code changes - when code is pushed to the repository
  - Time - specific time of the day
  - Webhook - when receive a request, eg HTTP POST

What is a CI/CD Pipeline?

- CI/CD pipeline is a list of task to be performed when the CI/CD process is triggered



https://wpblog.semaphoreci.com/wp-content/uploads/2019/03/cicd-pipeline-introduction.png

Major Stages in a CI/CD Pipeline

- Source
  - Pull the sources code and artifacts
- Build
  - Refers to building/compiling the source
  - Some runtime requires compilation some do not
    - Eg. Compiled - Java, Typescript, Golang
    - Eg. Not compiled - JavaScript, Python
  - Build or package artifacts eg images, JAR/WAR files, executables

Major Stages in a CI/CD Pipeline

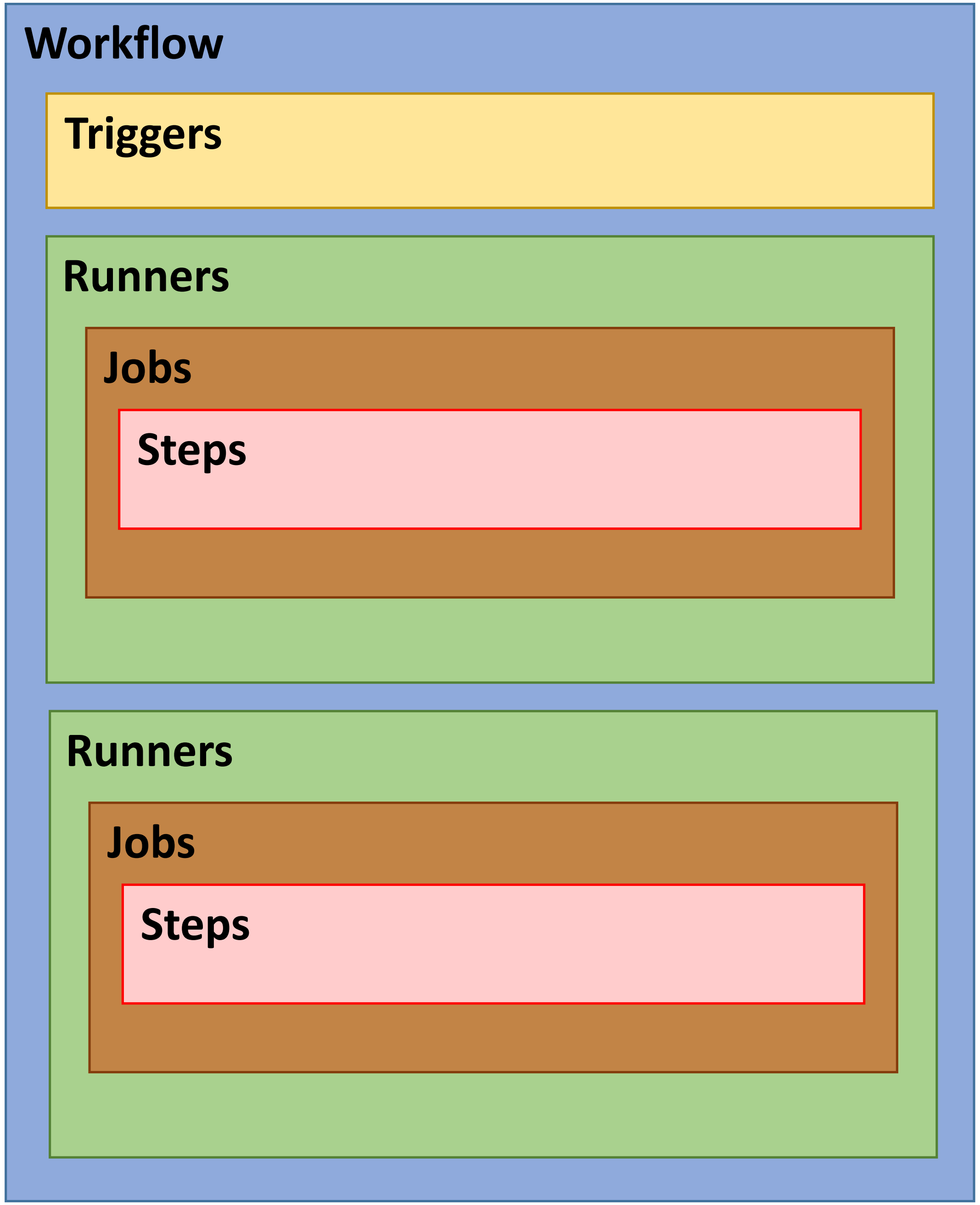
- Test
  - Run test to validate that the application is behaving correctly
  - Unit test, smoke test, integration test
  - May also generate coverage report to determine if we have written sufficient test to validate the software
  - Test may pass or fail depending on the test or coverage metrics
- Deploy
  - Applications are deployed to servers
    - Eg. uploaded to servers, deployed to Kubernetes or other container orchestration platform
  - Application may required infrastructure to be provisioned with Infrastructure as Code tools
    - Eg. Ansible, Terraform

Github Actions

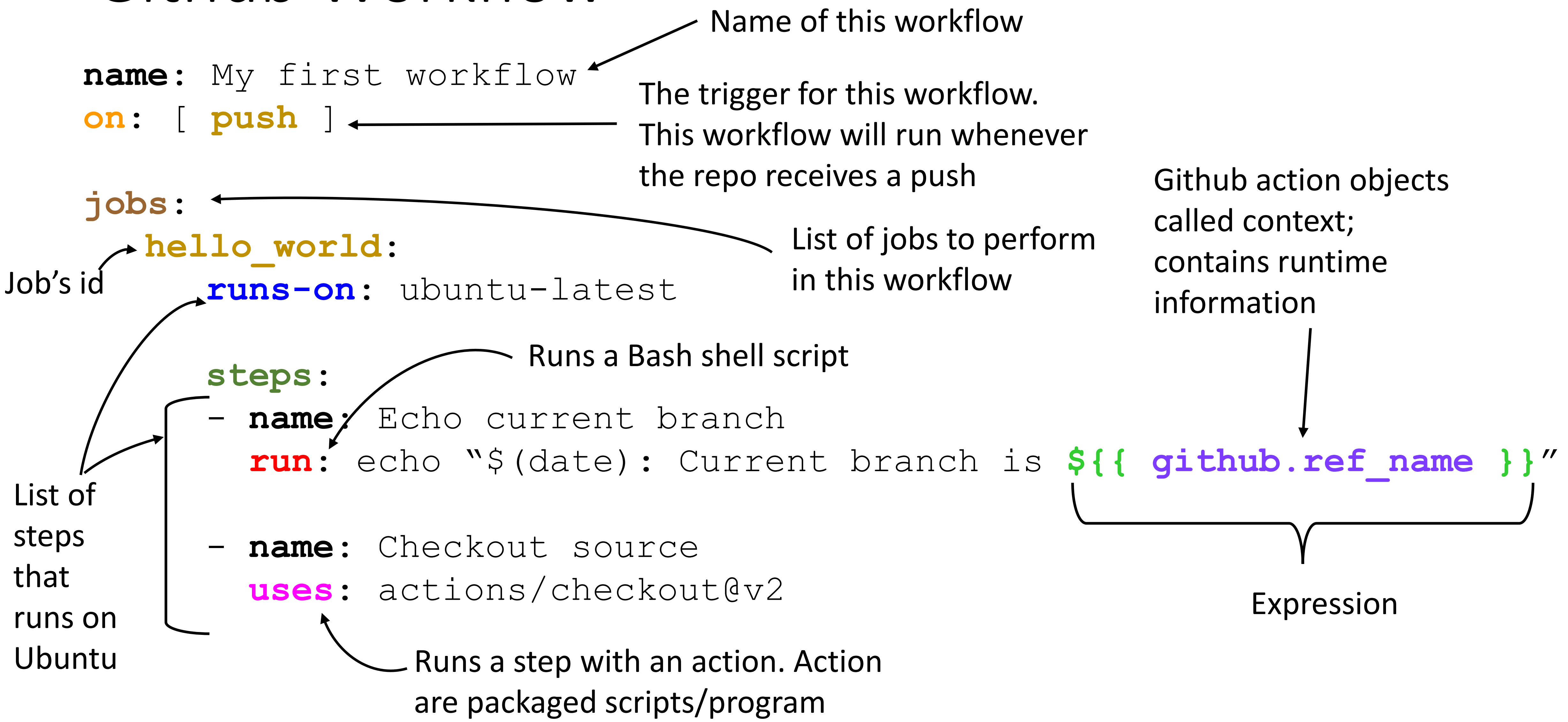
- Github Actions is tool within Github that automates software development and deployment workflow
  - A CI/CD platform
- Actions workflow are triggered by events
  - Eg. push to a branch, create a pull request, specific date/time or by dispatches (external events via webhooks)
- Workflows are created in `.github/workflows` directory
  - At the repository’s root

Workflow

- Github workflow files are written in YAML
  - Structurally very similar to JSON
  - Use indentation to indicate embedding like Python
- Can have multiple workflows in .github/workflows directory
  - Workflow will trigger according to the defined condition



Github Workflow



```
name: My first workflow
on: [ push ]

jobs:
   hello_world:
      runs-on: ubuntu-latest

      steps:
      - name: Echo current branch
        run: echo “$(date): Current branch is ${{ github.ref_name }}”

      - name: Checkout source
        uses: actions/checkout@v2
```

Name of this workflow

The trigger for this workflow. This workflow will run whenever the repo receives a push

List of jobs to perform in this workflow

Job’s id

Github action objects called context; contains runtime information

Runs a Bash shell script

Expression

List of steps that runs on Ubuntu

Runs a step with an action. Action are packaged scripts/program

Workflow Trigger

```
on: [ push ]

on:
   push:
```

- Workflow will trigger whenever there is a push to any branch

```
on:
   push:
      branches:
      - ‘feature/*’
      tags:
      - ‘v*’
      - ‘!v*-beta’
```

- The trigger is filtered whenever there is a push to any of the branches or tags with names that matches the specified pattern

List of event triggers:
https://docs.github.com/en/actions/using-workflows/events-that-trigger-workflows

Runners and Actions

- Runners is the environment which executes the workflow actions/steps
- List of runners
  - https://github.com/actions/virtual-environments#available-environments
- Runners provide a generic environment
  - Eg. An Ubuntu server
  - Need to setup the runner with one or more actions in steps

- Typical workflow phases
  - Setup the runner
    - Eg. setup application runtime
    - Eg. install required libraries
  - Pull source code into the runner
    - Either from the current repository
    - From other source
  - Build and/or test the application
  - Post build
    - Eg. deploy application once the test is successful
    - Eg. upload build artifacts

Example

```
jobs:
  build_and_deploy:
    runs-on: ubuntu-latest
    steps:
    - name: Setup nodejs
      uses: actions/setup-node@v3
      with:
        node-version: ‘17’
    - name: Checkout source
      uses: actions/checkout@v3
    - name: Install application dependencies
      run: npm ci
    - name: Build Angular application
      run: npm run build
    - name: Copy files to S3 bucket
      uses: BetaHuhn/do-spaces-action@v2
      with:
        access_key: ${{ secrets.access_key }}
        ...
```

List of steps to be performed on the runner

Setup the runtime

Checkout source from rep

Install application dependencies

Compile the application

Copy the compiled artifacts to a production environment

Workflow Steps

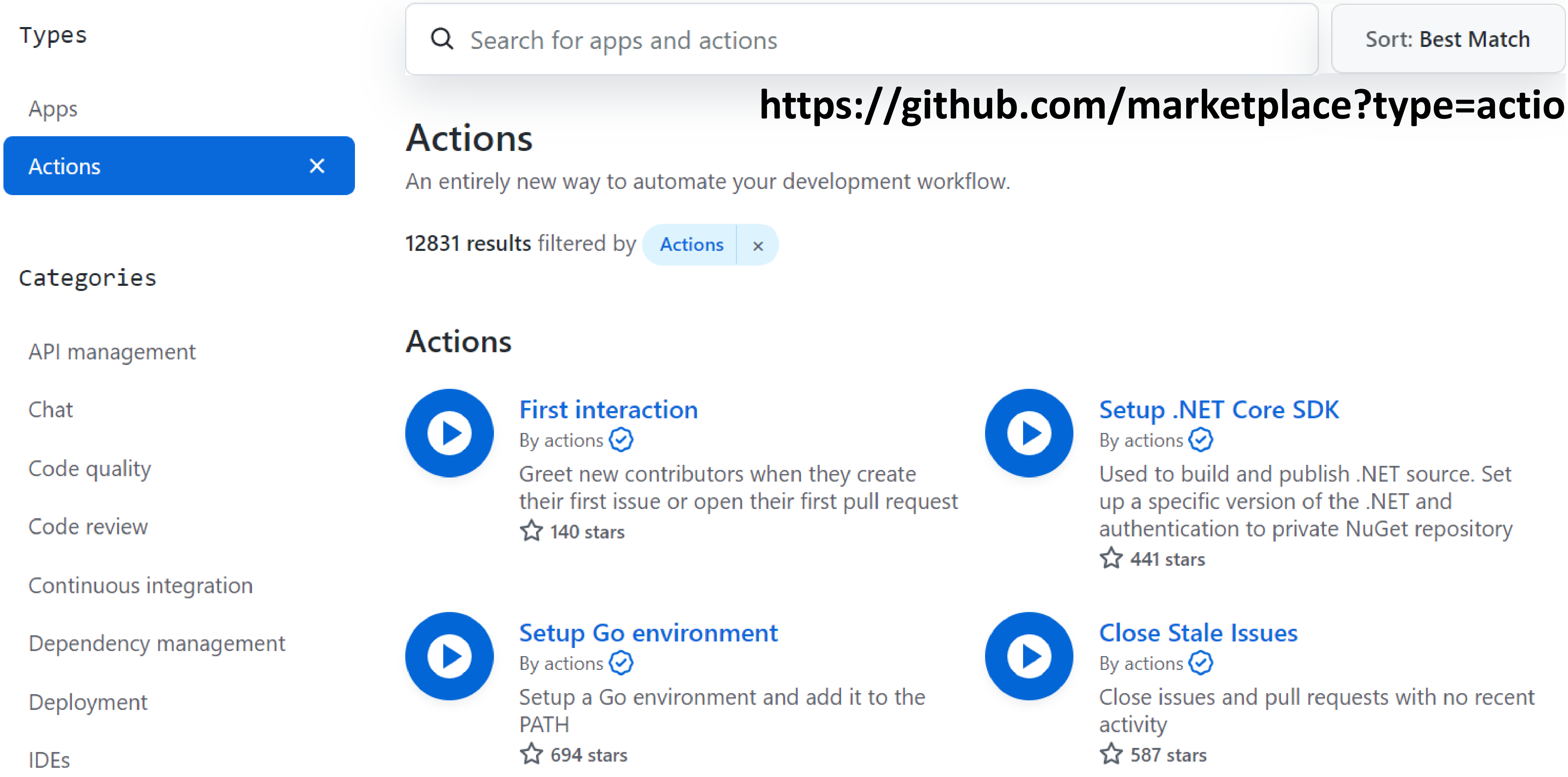
```
run: npm ci
```

- Run a Bash script on the runner
  - Runner in this case have to be a Linux based runner

```
uses: actions/setup-java@v2
with:
    java-version: ‘17’
    distribution: ‘temurin’
```

- Use a packaged script called actions
- Most actions will have configuration parameters
  - Configuration are passed into the action using the `with` attribute
  - Followed by a list of parameters

Github Action Marketplace



https://github.com/marketplace?type=actions

Types

Apps

Actions ×

Categories

API management

Chat

Code quality

Code review

Continuous integration

Dependency management

Deployment

IDEs

Search for apps and actions

Sort: Best Match

Actions

An entirely new way to automate your development workflow.

12831 results filtered by Actions ×

Actions

First interaction
By actions
Greet new contributors when they create their first issue or open their first pull request
☆ 140 stars

Setup .NET Core SDK
By actions
Used to build and publish .NET source. Set up a specific version of the .NET and authentication to private NuGet repository
☆ 441 stars

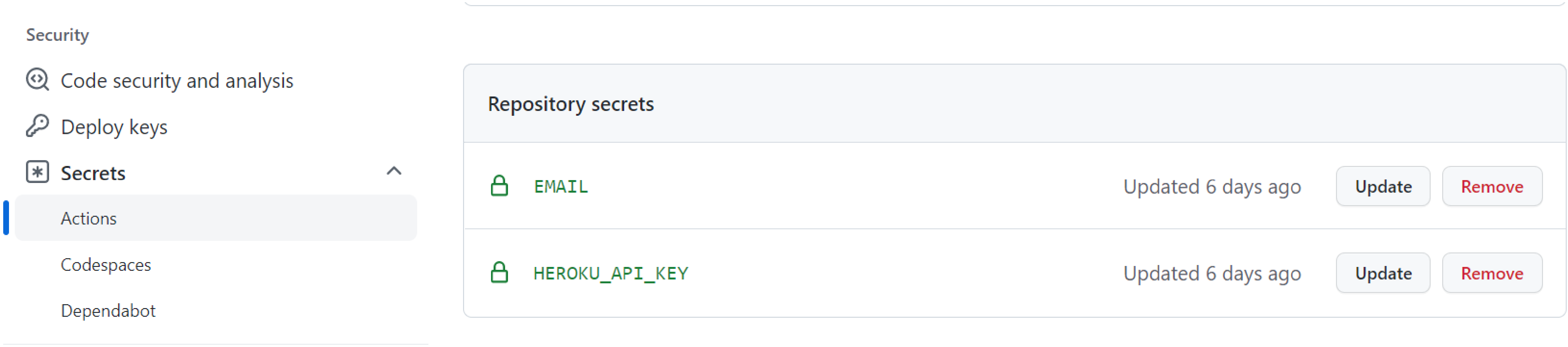
Setup Go environment
By actions
Setup a Go environment and add it to the PATH
☆ 694 stars

Close Stale Issues
By actions
Close issues and pull requests with no recent activity
☆ 587 stars

Secrets

- Some workflow requires password or secret token to work
  - Eg. deploying to cloud platform
  - Eg. pulling a private repository
- Need to be able to references these sensitive data without explicitly writing them in the workflow
  - Otherwise the password will be exposed
- Github actions allows you to defined these sensitive information as secrets in the repository
  - Workflow then references these secrets as expressions using the `secrets` context

Adding Github Secrets



Repository secrets

EMAIL	Updated 6 days ago	Update	Remove
HEROKU_API_KEY	Updated 6 days ago	Update	Remove

Use secrets in workflow to protect sensitive data

```
- name: Deploy
  uses: akhileshns/heroku-deploy@v3.12.12
  with:
    heroku_api_key: ${{ secrets.heroku_api_key }}
    heroku_app_name: myapp
    heroku_email: ${{ secrets.email }}
    branch: ${{ github.ref_name }}
```