**Use cases**

Lets say there’s a bug, we make a separate branch, fix it and make a pull request. The PR is tested and then merged into main. Perhaps you want to add a few tags at the same time

These are all workflows – repeated sequences of activities every time we want to update our main branch

When something happens in/to your repository, these are GitHub events

How github actions automates these workflows.

GA listens to these events. Each event triggers a workflow

Most common workflow example is a CI/CD pipeline

Using GA means that it is integrated into your code repo, rather than another 3rd party tool

Name = the name of the workflow

On = defines the events that trigger your workflow to run

Push = an ‘on’ event. Whenever code is pushed to your repo, this event is triggered.

Pull = another ‘on’ event, except for when the repo is pulled

Branches = the branches where we want to listen for this event

Jobs = the jobs of the workflow

Build = not a keyword in GA, but one of the jobs. It’s an id/label of the job, rather than being the actual jobs itself

Runs-on = what type of virtual machine the job will run on

Steps = the list of tasks that the job has to execute. Steps = tasks

Uses = allows you to run a reusable action within a step, like a pre written script. Prevents you from repeating yourself. Provides modularity

Under steps, each – name:, uses: …. is a step. The hyphen begins a new step

With = the input parameters for the provided action (“uses”)

Using = runtime environment of the action itself. The three possible values for using are nodeX, docker or composite.

Composite means that it bundles multiple steps into a single, reusable action

${{}} is an expression. It is how you access variables within your action file, specifically those that relate to the action like inputs and outputs

$GITHUB\_OUTPUT is a temp env var that is a filepath to all of the output variables for this action

Learning how to set up a composite action in GA and how to test it remotely

Action and workflow files can’t accept the same keywords