In this topic, you will learn to make a basic deploy pipeline with GitHub actions for your code: quickly and easily.

Adding actions to your own repository

To begin, you need to create a new GitHub repository. This repository should contain a text file, with the name

`hello.txt`

. Inside the file, paste the following line:

```
Hello world from github
```

Use

`mkdir .github/workflows`

or

`mkdir -p .github/workflows`

to create the workflow directory inside your local repository. To create workflows (configuration files used for pipelines) you need to create a YAML file. Let's name the workflow;

`hello_workflow.yaml`

.

Then paste the following template inside

`hello_workflow.yaml`

.

```
name: hello_workflow
```

```
on: [push]
jobs:
  print-hello:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: Hello world using echo
        run: echo "$(<hello.txt)"
      - name: Hello world using cat

        run: cat hello.txt
```

Now you can push your local repository to GitHub. The workflow file gets triggered when you push the code:

```
git add .
git commit -m "Added an initial Github Action"

git push
```

View logs

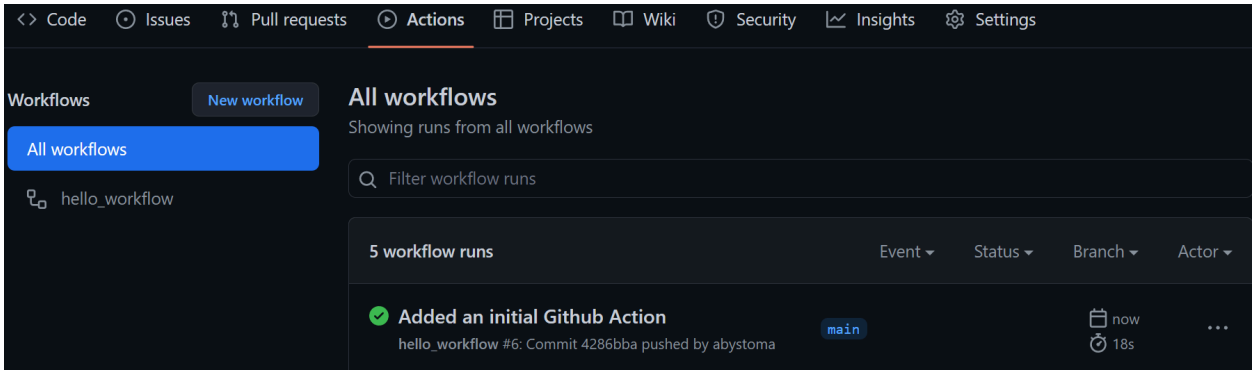
The GitHub action you created will start immediately after your event is executed. In your case the event is

`push`

. To see the workflow, go to the GitHub repository and select the

`Actions`

tab.



The icon before the commit name of your

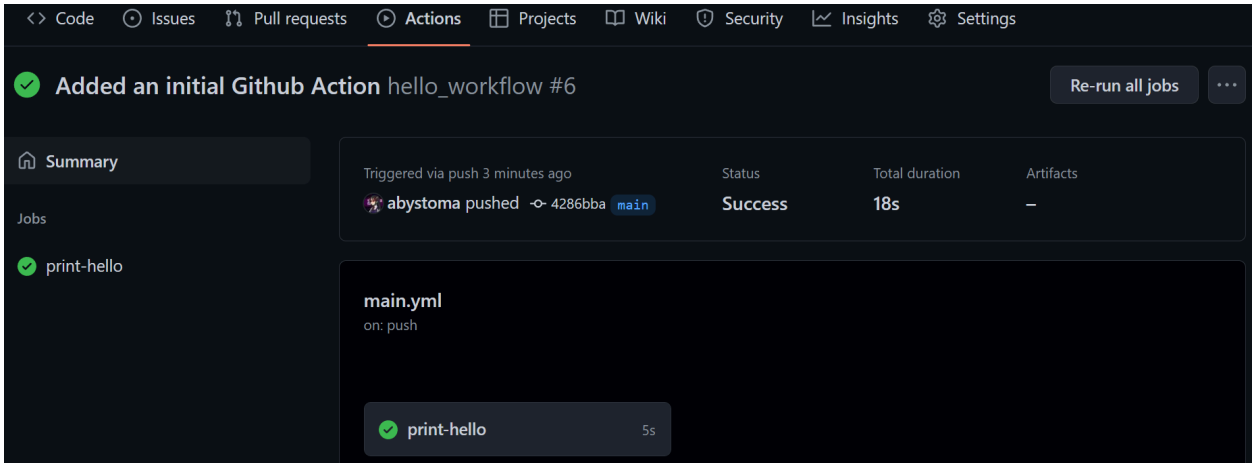
`hello_workflow`

workflow represents the current state of the workflow. The yellow circle means that the actions are still running. Once the actions are completed, a green tick will appear.

If you click on the commit name of your workflow

`hello_workflow`

, you will be able to see more information:

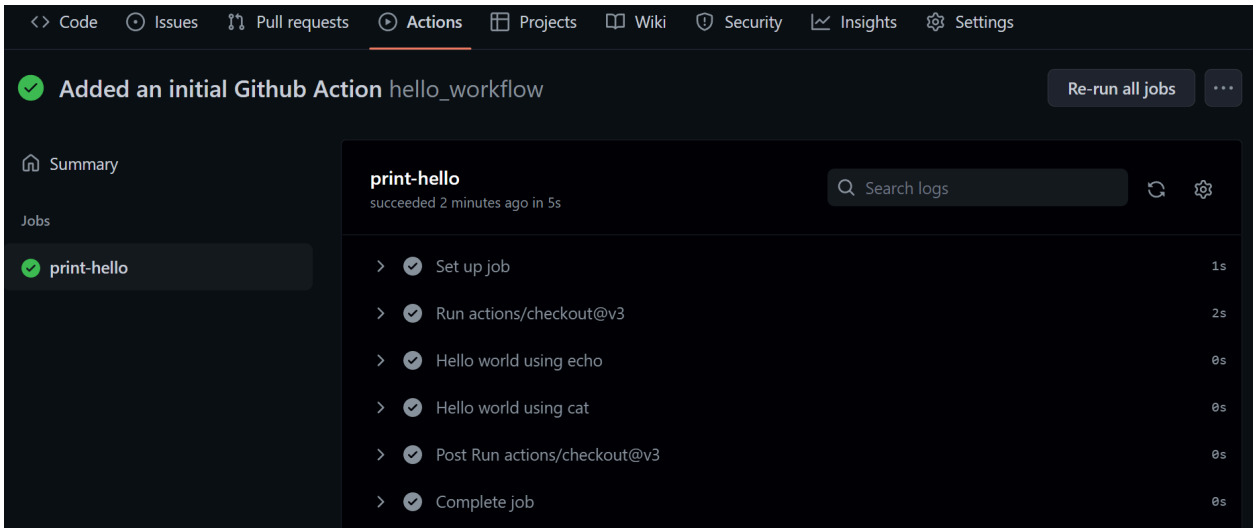


All the actions took 18 seconds to complete.

Click on the

`print-hello`

job in the workflow to see more information about the job.



You can see the different stages in the job. At first

`actions/checkout@v4`

step is executed. The role of this action is to check out your repository to the

`ubuntu-latest`

runner that will run your action. Then two more custom steps are executed which can be defined in your job, where you are printing the contents of the text file in your repository.

You can expand the actions by clicking on the triangle before the name of the action.

After expanding, you can see the output of your action.

The workflow file

The workflow file has several sections, which are explained below:

```
name: hello_workflow
```

The

`name`

keyword sets the name of the workflow. You can assign any suitable name that you prefer.

```
on: [push]
```

The

`on`

keyword specifies when the workflow will be executed. Your workflow here is executed when you push something into your repository.

```
jobs:
  print-hello:
```

Here you can list all the jobs you want to run by indenting under the

`jobs`

keyword. In this workflow, there is only one job called

`print-hello`

.

```
   runs-on: ubuntu-latest
```

The

`runs-on`

keyword is indented under the jobs name. This keyword specifies the virtual environment in which your workflow should run. Your workflow will run in the latest version of Ubuntu, as mentioned in the workflow file.

```
   steps:
      - uses: actions/checkout@v4
      - name: Hello world using echo
        run: echo "$(<hello.txt)"
      - name: Hello world using cat
        run: cat hello.txt
```

These are the steps in your

`print-hello`

job. Each step is preceded by

`-`

and is indented under the

`steps`

keyword. All the steps are executed sequentially. You can specify the name of the step with the

`name`

keyword, which will be logged as the step name once the workflow is completed.

The steps can be any command that you want to run. For example;

```
      run: cat hello.txt
```

This runs the command

`cat hello.txt`

(cat just prints the contents of the hello.txt file onto the screen).

You need to check out your repository to the runner so that your workflow can access the repository. This is the reason why you should start your job with this step:

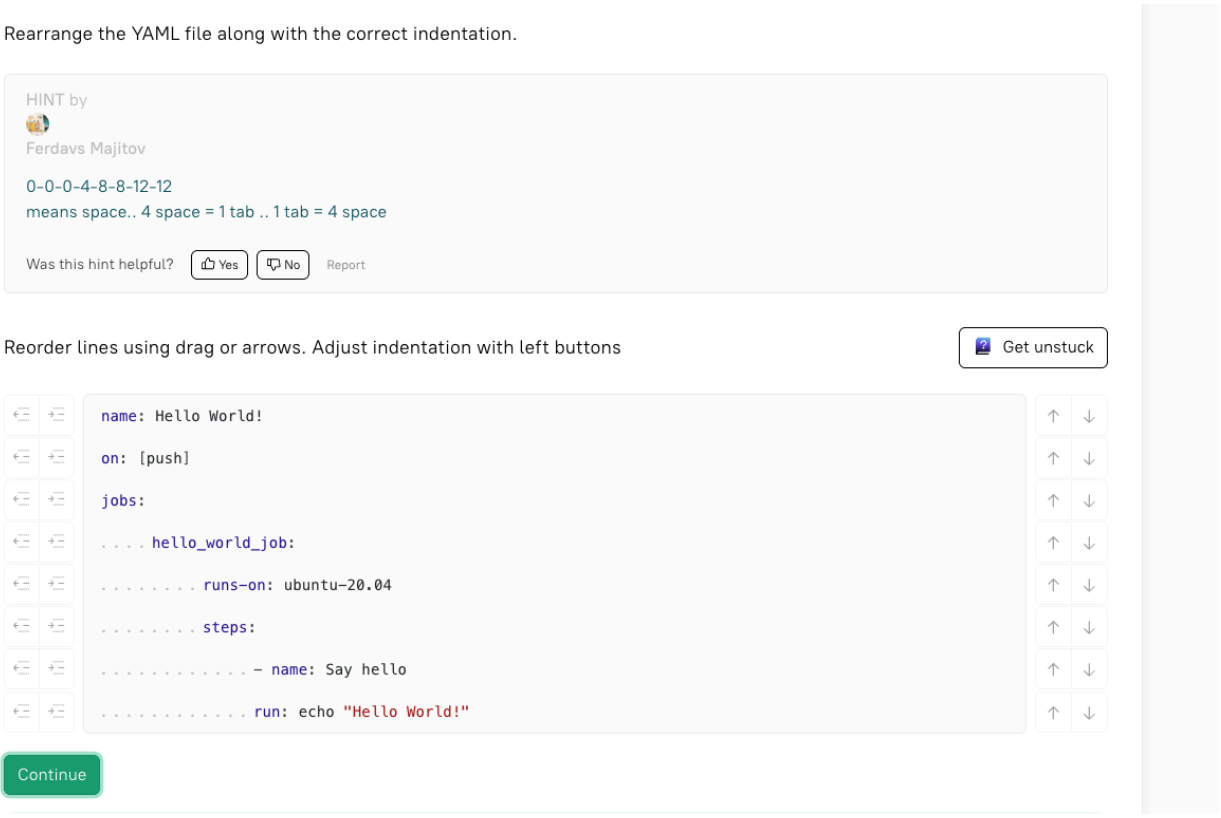
```
    - uses: actions/checkout@v4
```

`@v4`

means that version 4 of the checkout package should be used. You need to specify a version because packages are being continuously updated to newer versions which may introduce errors. So you should stick to one version of the package.

Conclusion

In this topic, you have learned to use GitHub actions for your repository. This simple workflow will help you deal with more complex workflows in the future.



Why is the

`actions/checkout@v3`

package used?

checks out the project source code from git

```yaml
name: hs_workflow_intro_to_github_actions
on: [push]
jobs:
  show-secret-phrase:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
        with:
          repository: abystoma/test
          ref: CombinedPhrase
      - name: Create file "hello.txt" with the word Hello
        run: ???
      - name: Show me the answer
        run: cat hello.txt from.txt
```

In the second step, write a bash command to create a text file `hello.txt` with the word `Hello` inside the file. Then run the workflow to find the combined phrase. Your task is to find the secret phrase produced from `hello.txt` and `from.txt` file combined. After successfully running the workflow, go to the `Actions` tab and click on the name of the workflow. Then you need to provide the output of the stage, "Show me the answer".