

1. How to Test GitHub Actions

Using nektos/act

For easy command-line local testing.

0. Note: see also companion document [2. Additional Notes for GitHub Actions](#)

1. Install [nektos/act](#)

- a. Requires docker to run
- b. Ensure correct installation by typing “act” into your VS Code terminal (or where you usually perform git commands)
 - i. This should run any workflows trigger “on: push”
 - ii. For hackforla/website, this would be the Lint SCSS workflow.
 - iii. The last two lines in terminal looks like this

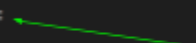
```
[Lint SCSS/Lint SCSS] ✓ Success - Main Lint SCSS  
[Lint SCSS/Lint SCSS] 🚩 Job succeeded
```

2. Determine what is required for the GHA of interest

- a. Read?
 - i. Any repo
 - ii. Ex. anything you can access just by looking at [https://api.github.com/repos/hackforla/website/issues/\[issue number\]](https://api.github.com/repos/hackforla/website/issues/[issue number])
- b. Write?
 - i. Only available in your own repo
 - ii. Ex. adding comments, labels, and other **actions**
- c. New issue or PR?

3. Manually trigger workflows with the command: act -j [name of workflow]

- a. [name of workflow] can be found here

```
jobs:  
  Add-Update-Label-to-Issues-Weekly-Testing:   
    runs-on: ubuntu-latest  
    steps:  
      - uses: actions/checkout@v2  
      - uses: actions/github-script@v4  
      env:  
        IN_PROGRESS_COLUMN_ID: ${{ secrets.IN_PROGRESS_COLUMN_ID }}  
      with:  
        script: |  
          const { IN_PROGRESS_COLUMN_ID } = process.env;  
          const script = require('./github-actions/add-update-label-weekly/add-label-test.js');  
          return script({ g: github, c: context }, IN_PROGRESS_COLUMN_ID);
```

- b. For this workflow, the command would be:
 - i. `act -j Add-Update-Label-to-Issues-Weekly-Testing`

4. Most likely, you'll run into this error

```
[Add Update Label to Issues Weekly/Add-Update-Label-to-Issues-Weekly] ! ::error::Unhandled error: Error: Input required and not supplied: github-token
[Add Update Label to Issues Weekly/Add-Update-Label-to-Issues-Weekly] ✖ Failure - Main actions/github-script@v4
[Add Update Label to Issues Weekly/Add-Update-Label-to-Issues-Weekly] exitcode '1': failure
[Add Update Label to Issues Weekly/Add-Update-Label-to-Issues-Weekly] 🚫 Job failed
Error: Job 'Add-Update-Label-to-Issues-Weekly' failed
```

- a. This workflow uses `actions/github-script@v4`, which requires a github token
- b. [actions/github-script@v4](#) is an easy way to access Github repos and information and is used in almost all of our GHAs

5. Create a personal access token

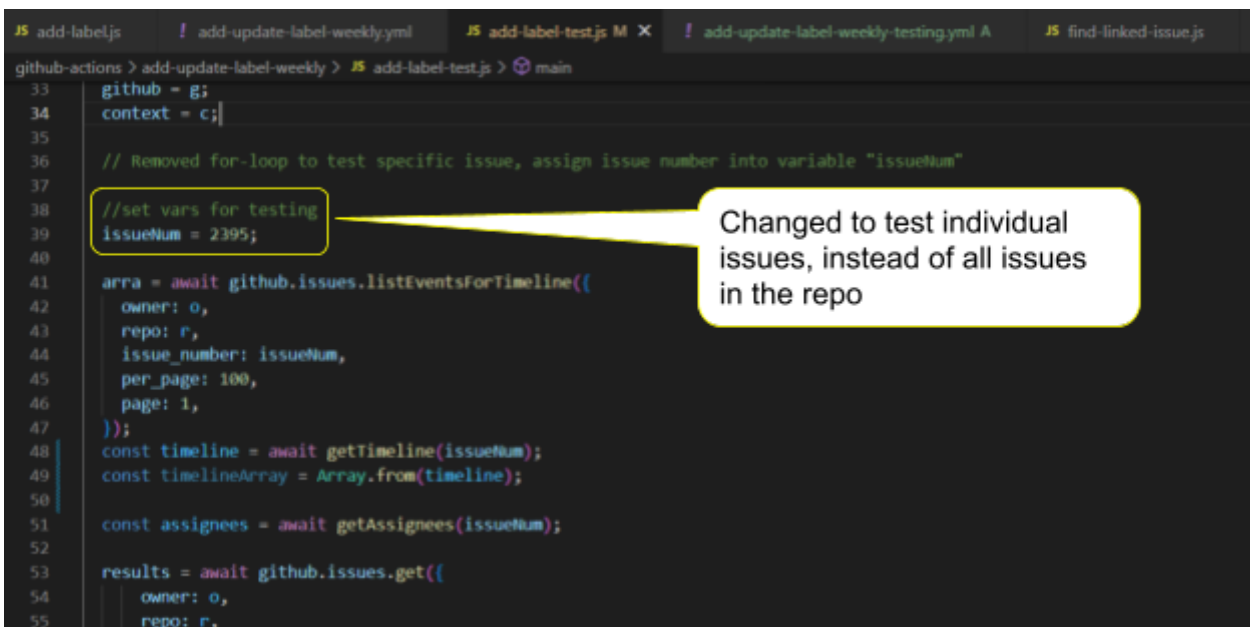
- a. Go to your [GitHub profile settings](#) >> [Developer settings](#)
- b. This is where you can create personal access tokens
 - i. Either fine grain or classic works
 - ii. Make sure to copy it

6. Pass your new token into the command

- a. `act -s GITHUB_TOKEN=[paste token] -j [title of workflow]`
- b. Run the command again and should be error free!

7. Make copies of files of interest and start making edits

- a. For example:



```
JS add-label.js | ! add-update-label-weekly.yml | JS add-label-test.js M X | ! add-update-label-weekly-testing.yml A | JS find-linked-issue.js
github-actions > add-update-label-weekly > JS add-label-test.js > main
33 | github = g;
34 | context = c;
35 |
36 | // Removed for-loop to test specific issue, assign issue number into variable "issueNum"
37 |
38 | //set vars for testing
39 | issueNum = 2395;
40 |
41 | arra = await github.issues.listEventsForTimeline({
42 |   owner: o,
43 |   repo: r,
44 |   issue_number: issueNum,
45 |   per_page: 100,
46 |   page: 1,
47 | });
48 | const timeline = await getTimeline(issueNum);
49 | const timelineArray = Array.from(timeline);
50 |
51 | const assignees = await getAssignees(issueNum);
52 |
53 | results = await github.issues.get({
54 |   owner: o,
55 |   repo: r,
```

```

else { // all the events of an issue older than fourteen days will be processed here
  // this chunk will likely be run first because index of 0 starts with oldest events
  if (checkCurrentPR(moment,assignees)) { // checks if cross referenced older than fourteen days
    console.log("last event is "+moment.event+ " by "+ moment.actor.login+ " on "+moment.created_at);
    console.log('found PR');
    return {result: false, labels: statusUpdatedLabel}
  }
}

```

Using console.log to check what variables hold

- b. The copies can be used as testing environments for PR reviewers to easily test the action locally

8. Need an event for the action to run?

- a. You can manually trigger via step #6
- b. Use the event path option
- c. Create events in your forked repo

Tips:

- The console log can be difficult to find. It's tucked between lots of output.

```

Add Update Label to Issues Weekly Testing/Add-Update-Label-to-Issues-weekly-Testing] docker exec cmd-[node /var/run/act/actions/actions-github-scri
(node:19) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usability issues. Please use the Buffer.alloc(), Buffer.allocUnsafe()
(Use 'node --trace-deprecation ...' to show where the warning was created)
post 14 day event cross referenced
Add Update Label to Issues Weekly Testing/Add-Update-Label-to-Issues-weekly-Testing] ::set-output:: result="no updates needed for issue #2395 (GITHU
ect Board)"
Add Update Label to Issues Weekly Testing/Add-Update-Label-to-Issues-weekly-Testing] Success - Main actions/github-script@v4
Add Update Label to Issues Weekly Testing/Add-Update-Label-to-Issues-weekly-Testing] Job succeeded

```

- nektos/act could be:
 - Helpful for GHAs like 2 weeks inactive (instead of creating an issue in your own repo and waiting for it to age)
 - Difficult with GHAs that involve **actions** with Project Board (you'll probably have to recreate the Project Board in your own repo and populate with issues and PRs)
- HfLA GHAs fall into 2 main categories
 - Scheduled workflows
 - Use act -s GITHUB_TOKEN=[paste token] -j [title of workflow]
 - Triggered workflows
 - Use act [] -e pull_request.json
- act is simply another tool at your disposal
 - If you can't directly test a function, there may be other indirect ways to test it
- Check out [https://api.github.com/repos/hackforla/website/issues/\[issue number\]](https://api.github.com/repos/hackforla/website/issues/[issue number]) and its urls