

Introduction to GitHub Actions

Scott Burns

Penny University Chat
2019-02-05

Caveats

- I don't work/speak for GitHub
- Actions is Beta Software
- Some info from emails w/ [support@github](mailto:support@github.com)

Table of Contents

- GitHub Webhooks
- External platforms/services/things
- Intro to Actions
- Things we've implemented
- Sore spots/excitement

**You can't understand the
scope of Actions w/o
understanding Webhooks**

What are webhooks?

- Emails are one form of notification GH sends to alert you to activity on a repo. (But at some point you've probably turned them off)
- Webhooks are the machine-friendly system GH uses to notify other systems of events.
- GH POSTs to an endpoint of your choosing with JSON payload describing the event.

**An event is literally
any activity in GitHub**

**There are a lot of
types of events**

CheckSuiteEvent
MembershipEvent
CommitCommentEvent
SecurityAdvisoryEvent
ContentReferenceEvent
ProjectColumnEvent
CreateEvent
PageBuildEvent
DeleteEvent
DeploymentStatusEvent
OrganizationEvent
MemberEvent
RepositoryImportEvent
InstallationRepositoriesEvent
MarketplacePurchaseEvent
RepositoryVulnerabilityAlertEvent
WatchEvent
TeamAddEvent
CheckRunEvent
TeamEvent
FollowEvent
ProjectCardEvent
ForkApplyEvent
DownloadEvent
GistEvent
InstallationEvent
GollumEvent
ProjectEvent
StatusEvent
OrgBlockEvent
ReleaseEvent
PullRequestReviewCommentEvent
RepositoryEvent
GithubAppAuthorizationEvent
PullRequestReviewEvent
MilestoneEvent
PushEvent
IssuesEvent
IssuesCommentEvent
DeploymentEvent
PullRequestEvent
PublicEvent

[\(https://developer.github.com/v3/activity/events/types/\)](https://developer.github.com/v3/activity/events/types/)

Configuring Webhooks

- Set at organization-level (for all repos owned by org) or repo-level
 - <https://github.com/organizations/{org}/settings/hooks>
 - <https://github.com/{org}/{repo}/settings/hooks>
- Opt-in to particular events or get the firehose
- Typically PushEvent is the most important. Represents a user pushing one or more commits to a repository.

✓ All Systems Operational

Current status

Git Operations ?	✓ Operational
API Requests ?	✓ Operational
Issues, PRs, Dashboard, Projects ?	✓ Operational
Notifications ?	✓ Operational
Gists ?	✓ Operational
GitHub Pages ?	✓ Operational

Webhooks are important to GitHub

githubstatus.com

External Services Are Built on Webhooks

- Jenkins (with plugin)
- Travis-CI
- Any external service that is automagically doing *things* (CI/linting/testing/codecov/etc) is integrated into GitHub via webhooks

Scripting Interfaces

- Each of these external services lets us tell them what to run.
 - Jenkinsfile
 - .travis.yml
- We use these files to tell those systems what to run for particular events.

request's .travis.yml

```
sudo: false
language: python
# command to install dependencies
install: "make"
# command to run tests
script:
  - |
    if [[ "$TRAVIS_PYTHON_VERSION" != "2.6" ]] ; then make test-readme;
fi
  - make ci
cache: pip
jobs:
  include:
    - stage: test
      script:
        - make test-readme
        - make ci
      python: '2.7'
    - stage: test
      script:
        - make test-readme
        - make ci
      python: '3.4'
  ...
```

<https://github.com/requests/requests/blob/master/.travis.yml>

Cool, right?

- We can script these external services to do the same thing every time anybody pushes to our repo.
- You might implement Continuous Integration/Continuous Deployment on top of these events:
 - For example, these services will often use GitHub's Checks API to provide build status for each commit they receive.
 - And then you get the nice green check on your commits.
 - And can gate PRs to ensure only those with passing tests get merged into the main branch.

cool cool cool

Warning:
Pure Speculation
incoming

If I'm GitHub, why would I
want *my users* paying external
services to *just* run scripts?

**Products naturally
evolve this way**

Thus (maybe?) born
GitHub Actions

GitHub Actions

Workflows describe a tree of actions to run in response to a webhook

GitHub Actions

Actions describe scripts that
run in Docker containers

Yeah yeah tl;dr?

- We declare workflows in `./.github/main.workflow`
- As activity occurs in GH and is delivered as a webhook, Actions resolves the tree of actions for workflows bound to that event and executes the necessary actions
- Bingo bango bongo, automation without having to run/maintain/pay* for CI infrastructure.

*Pricing on GitHub Actions is not public yet (<https://twitter.com/mcolyer/status/1083042907194482689>)

Example main.workflow

```
# ../github/main.workflow
```

Block "attributes"

```
workflow "continuous integration" {  
  on = "push"  
  resolves = ["test"]  
}
```

```
  action "lint" {  
    uses = "../actions/lint"  
  }
```

```
  action "test" {  
    needs = ["lint"]  
    uses = "../actions/test"  
  }
```

This is a "block"

(This is actually Hashicorp's HCL language)

Workflow Details


```
workflow "my workflow" {  
  on = "event type"  
  resolves = [  
    "First action",  
    "Second action"  
  ]  
}
```

- Version of `main.workflow` dependent on the event
 - Mostly at the SHA of the driving event, some events it may be at default branch
- Cannot drive further workflows from within a workflow; no endless cycles :)

Action Blocks

```
action "my action" {  
  needs = [  
    "Multiple",  
    "Successful",  
    "Actions"  
  ]  
  uses = "./actions/dir" OR  
  uses = "org/repo/dir@ref" OR  
  uses = "docker://[host]/image.tag"  
  env = {  
    ENV_VAR = "value"  
    ANOTHER_VAR = "else"  
  }  
  runs = "/entrypoint.sh"  
  args = [  
    "passed",  
    "as",  
    "list"  
  ]  
  secrets = ["SEKRET", "SHHH"]  
}
```

**Directory contains a Dockerfile,
Actions runtime builds it and
executes**



**Like above, public repository
for shared actions**



Public image repository

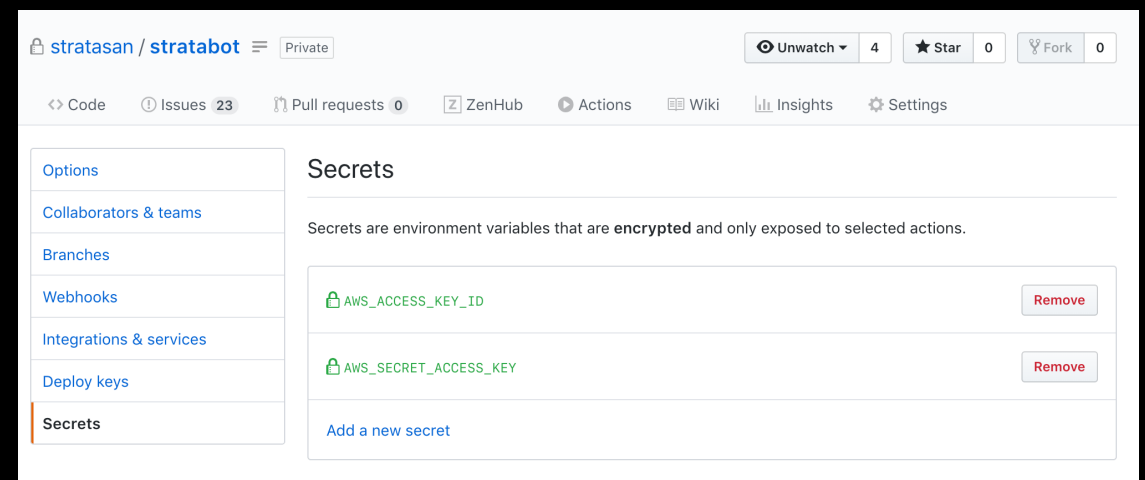


Runtime environment

- Actions get a CPU, ~4GB RAM, 100GB writable disk and network
- Repo checked out to SHA of event is accessible at /github/workspace (aka \$GITHUB_WORKSPACE)
- Event JSON written to \$GITHUB_EVENT_PATH
 - Have you met my friend jq...

Secrets

- Write-only UI in Settings
- Read-only as env vars for actions that specify them by name
- \$GITHUB_TOKEN is provided to make API requests to the GitHub API within an action.
- Careful not to log them...



Current workflows

- Pushes on stratasan/health get front-end linted and tested and python code is checked for black styling.
- Pushes to master on stratasan/engineering gets the site built and sync'd to the S3 bucket. Continuous Deployment ftw!
- Pushes to stratasan/stratabot are tested, the image is built, tagged and pushed to our Elastic Container Repository

(Our) Generic Actions

- stratasan/actions is a repo for some shared actions I've begun to build out. We're using them in places.
- Coming from Jenkins, I want to run a list of shell commands with a specific interpreter (python, node, etc) available.
- I find this makes `main.workflow` more readable

Community Actions of note

- actions/aws/cli
 - Pre-installed AWS CLI, will need AWS_ secrets.
- actions/docker/cli
- actions/bin/filter
 - Only continue subsequent actions if event matches a filter

Sore spots

- Running other containers in a network
 - To test health, we need a Postgres & Redis container. Jenkins currently uses a deprecated feature of Docker to do this (`--link` for those interested).
 - Canceled actions -> Failed PR checks :(ul> - Makes actions/bin/filter less useful right now
- Not-live logs as actions run
- UI isn't quite there yet, but it will improve

But the upside!

- Scriptable GitHub! (@jessfraz's words, not mine)
- And not just push! Basically all events! Which is wild!
 - Create/delete branches (build/teardown a staging environment just for that branch? Uh yes plz)
 - Deployments (visible deployments in a PR timeline)
 - Issues comments created/updated/deleted
 - Membership to a repo!
- No separate infra for CI :tada: :partyparrot:
 - Not so much the cost in AWS as "oh \$deity we need a Jenkins expert to sort that one out"

Further Reading

- <https://developer.github.com/actions/>
 - Dense. Don't skip sections
- <https://blog.jessfraz.com/post/the-life-of-a-github-action/>
- The open sourced actions in github.com/actions
 - I actually learned a lot re: shell scripting here