H1 # Github Actions

H2 Important Links

- Freecodecamp - What are Github Actions
- Linkedin Learning - Automating with Github Actions
- Github Marketplace - Using Add & Commit
- Github Marketplace
- My own - Github actions sandbox repo

H2 What are Github Actions ?

Actions are a relatively new feature to Github that allow you to set up CI/CD workflows using a configuration file right in your Github repo.

Previously, if you wanted to set up any kind of automation with tests, builds, or deployments, you would have to look to services like - Jenkins Automation tool, Circle CI and Travis or write your own scripts. But with Actions, you have first class support to powerful tooling to automate your workflow.

H2 Use-cases

- Automate Unit Tests
- Automate Build Pipelines
- Code Formatting
- Remote Deployments
- CI / CD
- Event based triggers

H2 How are they different from Jenkins ?

- **Jenkins** creates workflows using Declarative Pipelines, which are similar to **GitHub Actions** workflow files.
- **Jenkins** uses stages to run a collection of steps,
- **GitHub Actions** uses jobs to group one **or** more steps **or** individual commands.
- both **Jenkins** and **GitHub Actions** support container-based builds.

H2 Pros & Cons

H3 Pros

- Built in Support into Github
- Free to use* - (also contains premium pipelines Circle CI & Travis are premium) but simple deployment scripts are free to use
- Supports both CI & CD
- Huge Market place - Github Marketplace
- Platform Agnostic - but supports a wide variety of Languages -
- Built in Container - Docker
- Can build your own custom pipelines and contribute to the community

H3 Cons

- They keep *"upgrading"* and changing shit up
- Has a bit of a steep learning curve
- YAML !! 🤔
- BOILERPLATE. Most of the time, It's just a copy paste 😄
- Some errors are harder to debug

H2 For Online Sales - Mid-Tier

Right now, we have multiple repos which require different **Wildfly Configuration**, with most containing **sensitive credentials**. While we can store these configuration directly in the Git repo. These can be considered as a **security risk**

H3 What if we use Github actions to store ONLY under certain branches ?

I propose to use the following 2 nice features of Github Actions

1. Storing these Wildfly Configuration in the repo ONLY under certain branches
2. Using Github Secrets to Obfiscate sensitive information

H3 1. Storing these Wildfly Configuration in the repo ONLY under certain branches

Github actions can be TARGETTED to run for either -

- Git Push actions
- Pull Requests

They can also be TARGETTED to run for certain branches - For Eg: `dev`

The following YAML script, creates a Custom file `env.txt` under the folder - build/env.txt

```yaml
name: Create custom file
on: push

jobs:
  run:
    name: Create custom file
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repo
        uses: actions/checkout@v2

      - name: Create file
        run: |
          cat <<EOT >> build/env.txt
          line 1
          line 2
          EOT
      - name: Commit changes
        uses: EndBug/add-and-commit@v7
        with:
          author_name: Pramod AJ
          author_email: avj2352@gmail.com
          message: 'Testing commit'
          add: '*.txt'
```

H3 2. Combining with Github Secrets

Combining the Above secrets make these Github Actions really powerful

H3 i. How to create Secrets

- Goto to Github Profile > Settings > Secrets
- Secrets need to camel-cased
- Values once entered, CANNOT be read - Hidden
- Values once entered, can only be overridden OR deleted, cannot be read / seen.
- **NOTE:** These secrets can be referenced in Github actions
- Secrets can also be based on Environment (making them a little dynamic)
- to access a Secret into a Github Actions

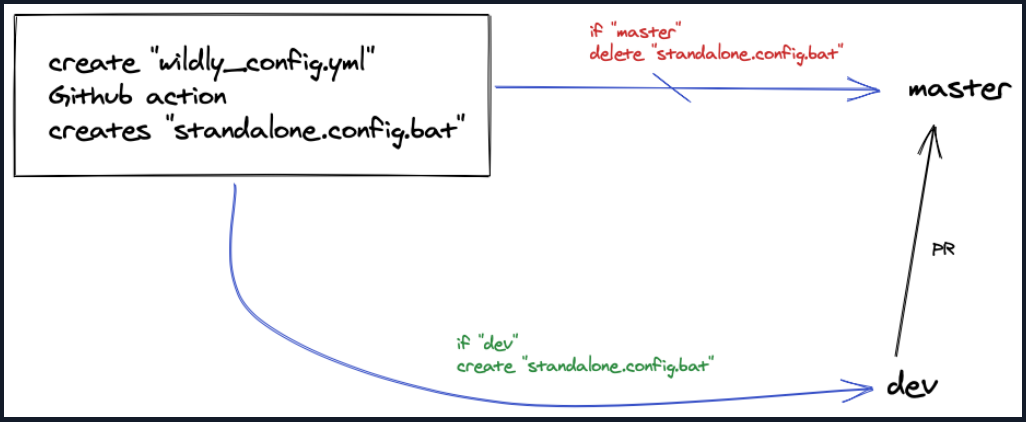
```
${{secrets.API_KEY}}
```

So the same above YAML script can be re-configured as follows:

```yaml
name: Create custom file
on: push
```

```yaml
jobs:
  run:
    name: Create custom file
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repo
        uses: actions/checkout@v2

      - name: Create file
        run: |
          echo -n "" > build/env.txt
          cat <<EOT >> build/env.txt
          line ${{secrets.DEV_NAME}}
          line ${{secrets.VERSION}}
          EOT
      - name: Commit changes
        uses: EndBug/add-and-commit@v7
        with:
          author_name: Pramod AJ
          author_email: avj2352@gmail.com
          message: 'Testing commit'
          add: '*.txt'
```



H3 Rules for Naming your Secrets

The following rules apply to secret names:

- Secret names can only contain alphanumeric characters (`[a-z]` , `[A-Z]` , `[0-9]`) or underscores (`_`). Spaces are not allowed.
- Secret names must not start with the `GITHUB_` prefix.
- Secret names must not start with a number.
- Secret names are not case-sensitive.
- Secret names must be unique at the level they are created at. For example, a secret created at the environment level must have a unique name in that environment, a secret created at the repository level must have a unique name in that repository, and a secret created at the organization level must have a unique name at that level.

- If a secret with the same name exists at multiple levels, the secret at the lower level takes precedence. For example, if an organization-level secret has the same name as a repository-level secret, then the repository-level secret takes precedence. Similarly, if an organization, repository, and environment all have a secret with the same name, the environment-level secret takes precedence.

To help ensure that GitHub redacts your secret in logs, avoid using structured data as the values of secrets. For example, avoid creating secrets that contain JSON or encoded Git blobs.

H2 Conclusion

Github Actions really shine when integrating CI / CD with AWS remote deployments / Docker deployments / Kubernetes Deployments.

The above code snippets is a VERY WEAK use-case to what Github action are really capable of 😄

H2 Trigger Github Actions on specific branches

- Dev: **create_env.yaml** file

  The following scripts, create a **default.json** environment yaml file while creating a PR to the **dev** branch.

- Master: **remove_env.yaml** file

  The following scripts, removes **default.json** or **development.json** environment yaml files while creating a PR to the **master** / **main** branch.

H3 i. Create Environment file on push to dev

The **create_env.yml** creates the following Environment file using values from the Repository Secrets

```
# Create a default.json environment file under backend/config/ folder
# Add Environment Variables from Repository Secrets
# Triggers for any push to the dev branch
name: Create env file
on:
  push:
    branches:
      - dev

jobs:
  run:
```

```
    name: Create env file
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repo
        uses: actions/checkout@v2

      - name: Create file
        run: |
          echo -n "" > backend/config/default.json
          cat <<EOT >> backend/config/default.json
          {
            "server": {
              "port": ${{secrets.SERVER_PORT}}
            },
            "google": {
              "client_id": "${{secrets.GOOGLE_CLIENT_ID}}",
              "client_secret": "${{secrets.GOOGLE_CLIENT_SECRET}}",
              "login_dialog_uri": "",
              "oauth_redirect_uri": "${{secrets.OAUTH_REDIRECT}}",
              "access_token_uri": "",
              "scopes": [
                "https://www.googleapis.com/auth/plus.login",

"https://www.googleapis.com/auth/plus.profile.emails.read"
              ],
              "response_type": "code",
              "grant_type": "authorization_code"
            },
            "jwt": {
              "secret": "${{secrets.JWT_SECRET}}",
              "expiry": 10
            },
            "db": {
              "uri": "${{secrets.DB_URI}}"
            }
          }
          EOT
      - name: Commit changes
        uses: EndBug/add-and-commit@v7
        with:
          author_name: Pramod AJ
          author_email: avj2352@gmail.com
          message: 'Adding Environment file'
          add: '*.json'
```

H3 ii. Remove Environment file on push to master / main

> *NOTE: as of the latest github version, by default creates "main" branch instead of the "master" branch*

The **remove_env.yml** simply removes any *.json files from the config folder

```yaml
# Remove default.json environment file under backend/config/ folder
# Remove Environment files - default.json & development.json
# Triggers for any push to main / master branches
name: Remove env file
on:
  push:
    branches:
      - master
      - main

jobs:
  run:
    name: Remove env file
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repo
        uses: actions/checkout@v2

      - name: Remove files
        run: |
          rm -rf backend/config/default.json
          rm -rf backend/config/development.json
      - name: Commit changes
        uses: EndBug/add-and-commit@v7
        with:
          author_name: Pramod AJ
          author_email: avj2352@gmail.com
          message: 'Removing Environment file'
          add: '*.json'
```
