Anleitung zum Aufsetzen einer einfachen Pipeline mit GitHub-Actions

Die Folgende Anleitung soll dazu dienen eine Pipeline mit GitHub-Actions für das Projekt in moderne Programmierkonzepte aufzusetzen. Ein Beispiel-Repository mit einer funktionierenden Pipeline ist unter folgendem Link zu finden:

https://github.com/STAT1C-Sy/mp_abschlussaufgabe

Damit das Aufsetzen funktioniert muss sich der Code in einem GitHub-Repository befinden: siehe Anleitung aus SE letztes Semester oder https://docs.github.com/en/github/getting-started-with-github/create-a-repo

Zur Wiederholung aus dem Vortrag befindet sich hier eine gute Liste mit den wichtigsten Begriffen: https://docs.github.com/en/actions/learn-github-actions/introduction-to-github-actions

Zum Aufsetzen müssen nur folgende Schritte durchgeführt werden:

1. Anlegen der Verzeichnisse /.github/workflows/ im Hauptordner des Projekts
2. Anlegen einer .yml Datei im workflows-Ordner (GitHub wird alle Workflows aus diesem Ordner parallel ausführen)
3. Einfügen von folgendem Code in die yml-Datei

```yaml
name: first-github-action
on: [push]
jobs:
  execute-tests:
    runs-on: ubuntu-latest
    steps:
      - name: checkout repo content
        uses: actions/checkout@v2
      - name: setup python
        uses: actions/setup-python@v2
        with:
          python-version: 3.8
      - name: execute tests
        run:
          python tests.py
```

Erklärung zum obenstehenden Code:

Zeile 1: „name: “ gibt den Namen des Workflows an

Zeile 2: „on“ gibt an wann der Workflow ausgeführt werden soll. Eine Liste der Events findet sich hier: https://docs.github.com/en/actions/reference/events-that-trigger-workflows

Zeile 3: „jobs“ Liste aller Jobs, die ausgeführt werden sollen

Zeile 4: Hier steht der Name des Jobs, in diesem Fall „execute-tests“. Der Name kann frei gewählt werden.

Zeile 5: „runs-on“ gibt an unter welchem Betriebssystem der Runner die Pipeline ausführen soll.

Zeile 6: „steps“ Liste der Schritte, die in diesem Job ausgeführt werden sollen, jeder Job hat einen Namen und ggf. zusätzliche Befehle, wie z.B. use oder run. Hier kann der Name auch wieder frei gewählt werden.

Step Nr.1 (checkout repo content): Dieser Befehl wird benötigt, damit der Job auf den Code in unserem Repository zugreifen kann, dazu wird eine von GitHub vordefinierte Action verwendet, die in „uses“ spezifiziert wird.

Step Nr. 2 (setup python): Initialisiert einen Python-Interpreter im aktuellen Job, der dann dazu genutzt werden kann, die tests.py Datei auszuführen. Hierzu wird auch wieder eine vordefinierte Action verwendet, der mit „with“ und dem Parameter „python-version“ mitgeteilt wird welche Python-Version verwendet werden soll.

Setp Nr. 3 (execute tests): führt den Befehl nach „run“ aus, in diesem Fall „python tests.py“, dadurch wird die tests.py Datei regulär ausgeführt.

4. Alle Änderungen mit git add . hinzufügen
5. Einen lokalen commit mit git commit -m „added actions pipeline“ machen
6. Die Änderungen mit git push auf den aktuellen Branch pushen
7. Wenn alles funktioniert hat, sollte der Workflow im Actions-Tab des Repositorys zu sehen sein: