# Github Actions

## GitHub Logos and Usage · GitHubWas ist Github?

GitHub ist eine Web-basierte Plattform für Versionskontrolle und Kollaboration, die auf Git basiert. Git ist ein verteiltes Versionskontrollsystem, das es Entwicklern ermöglicht, Änderungen an einem Code-Repository zu verfolgen, zu speichern und zusammenzuführen. Die Plattform bietet auch Funktionen für Bug-Tracking, Projekt-Management, Wiki-Dokumentation und kontinuierliche Integration und Bereitstellung.

## Was sind Github Actions?

GitHub Actions sind eine Funktion von GitHub, mit der Entwickler Workflows für ihre Softwareprojekte automatisieren können. Diese Workflows können aus mehreren Schritten bestehen, wie z.B. dem Kompilieren von Code, dem Ausführen von Tests, der Erstellung von Binärdateien oder der Bereitstellung von Code in einem Produktionsumfeld.

Die GitHub Actions sind in das GitHub-Repository integriert und können durch Ereignisse wie das Erstellen, das Erstellen von Pull-Anfragen oder das Markieren von Releases ausgelöst werden. Entwickler können auch benutzerdefinierte Ereignisse erstellen, um Workflows manuell oder programmgesteuert auszulösen.

## Warum Github Actions?

Github Actions ist eine leistungsstarke, integrierte CI/CD-Lösung, die in Github integriert ist und es Entwicklern ermöglicht, ihre Workflows direkt in ihrem Code-Repository zu definieren und auszuführen. Hier sind einige Gründe, warum Github Actions über Alternativen bevorzugt werden sollte:

1. Nahtlose Integration mit Github: Github Actions ist nahtlos in Github integriert, was es den Entwicklern ermöglicht, Workflows direkt in ihrem Code-Repository zu definieren und auszuführen. Dies vereinfacht den Entwicklungsprozess und reduziert die Notwendigkeit, zwischen verschiedenen Tools und Plattformen zu wechseln.
2. Flexibilität und Skalierbarkeit: Github Actions bietet eine Vielzahl von Optionen und Konfigurationen, die es Entwicklern ermöglichen, ihre Workflows flexibel zu gestalten und an die Bedürfnisse ihres Projekts anzupassen. Es kann auch nahtlos skaliert werden, um den Anforderungen von Projekten jeder Größe gerecht zu werden.
3. Große Auswahl an vordefinierten Aktionen: Github Actions bietet eine breite Palette von vordefinierten Aktionen, die häufig in der Entwicklung verwendet werden, wie z.B. das Bauen und Testen von Code, das Bereitstellen von Anwendungen und die Ausführung von Skripten. Diese Aktionen können einfach in den Workflow integriert werden, um Zeit und Aufwand bei der Entwicklung zu sparen.
4. Einfache Automatisierung: Github Actions kann genutzt werden, um eine Vielzahl von Aufgaben zu automatisieren, die Entwickler normalerweise manuell ausführen müssten. Dies kann dazu beitragen, den Entwicklungsprozess zu beschleunigen und die Fehleranfälligkeit zu reduzieren.
5. Kostenlose Nutzung für Open-Source-Projekte: Github Actions ist kostenlos für Open-Source-Projekte, was es Entwicklern erleichtert, die Vorteile der Plattform zu nutzen, ohne zusätzliche Kosten zu verursachen.

## Was für Bestandteile zählen zu den Github Actions?

1. Actions: Eine Action ist eine wiederverwendbare Sequenz von Befehlen, die ausgeführt werden können, um eine bestimmte Aufgabe in einem Workflow auszuführen. GitHub bietet eine Vielzahl von Aktionen, die Entwickler verwenden können, oder sie können ihre eigenen Aktionen erstellen und veröffentlichen.
2. Events: Events sind Aktionen, die innerhalb eines Repositorys ausgelöst werden, wie beispielsweise das Erstellen, Ändern oder Löschen von Code oder Pull Requests. Events lösen Workflows aus und können genutzt werden, um eine Automatisierung auszulösen.
3. Runner: Ein Runner ist eine ausführbare Anwendung, die GitHub Actions ausführt. Ein Runner kann auf einer beliebigen Plattform installiert werden und ermöglicht es Entwicklern, Workflows auf ihrer eigenen Infrastruktur oder in der Cloud auszuführen.
4. Job: Ein Job ist eine Sammlung von Aktionen, die in einer spezifischen Umgebung ausgeführt werden. Jeder Job ist Teil eines Workflows und kann abhängig von den Anforderungen des Workflows eine oder mehrere Aktionen enthalten.
5. Step: Ein Schritt ist eine einzelne Aktion, die innerhalb eines Jobs ausgeführt wird. Ein Job kann mehrere Schritte enthalten, die aufeinander aufbauen und aufeinander abgestimmt sind, um eine bestimmte Aufgabe auszuführen.
6. Workflow: Ein Workflow ist eine Sammlung von Jobs und Schritten, die zusammenarbeiten, um eine bestimmte Aufgabe auszuführen. Workflows werden durch Ereignisse ausgelöst und können auf bestimmte Branches, Tags oder Pull Requests beschränkt sein.

## Wie erstellt man solche Actions?

Zunächst muss ein Repository vorliegen, in welchem diese Action erstellt werden soll. Im Register Actions kann man dann einen neuen Workflow erstellen. Für diese Workflows gibt es je nach Inhalt des Repositorys auch geeignete Vorschläge bezüglich vorgefertigten Workflows. Hat man eine Vorlage gewählt, oder einfach eine leere neue Workflow Datei erstellt, so kann man diese nun nach seinen Bedürfnissen bearbeiten. Für mein Beispiel habe ich einen Workflow erstellt, wobei der Inhalt der Konfigurationsdatei so aussieht:

name: Manual workflow

on:

workflow\_dispatch:

inputs:

name:

description: 'Person to greet'

default: 'World'

required: true

env:

REGISTRY: ghcr.io

IMAGE\_NAME: ${{ github.repository }}

jobs:

build-and-push-image:

runs-on: ubuntu-latest

permissions:

contents: read

packages: write

steps:

- name: Checkout repository

uses: actions/checkout@v3

- name: Log in to the Container registry

uses: docker/login-action@f054a8b539a109f9f41c372932f1ae047eff08c9

with:

registry: ${{ env.REGISTRY }}

username: ${{ github.actor }}

password: ${{ secrets.GITHUB\_TOKEN }}

- name: Extract metadata (tags, labels) for Docker

id: meta

uses: docker/metadata-action@98669ae865ea3cffbcbaa878cf57c20bbf1c6c38

with:

images: ${{ env.REGISTRY }}/${{ env.IMAGE\_NAME }}

- name: Build and push Docker image

uses: docker/build-push-action@ad44023a93711e3deb337508980b4b5e9bcdc5dc

with:

context: .

push: true

tags: ${{ steps.meta.outputs.tags }}

labels: ${{ steps.meta.outputs.labels }}

Um kurz die Funktionalität dieses Workflows zu beschreiben: Wenn dieser Workflow erfolgreich durchläuft, wird automatisch ein Package dem Repository hinzugefügt, welches man sich pullen kann. Dieser Workflow ist manuell auszuführen, was durch „on: workflow\_dispatch“ konfiguriert wird. Im „env“ bereich kann man Variablen anlegen, welche in der yml verwendet werden können. Die richige Aktion wird dann im „jobs“ definiert, wobei man da die einzelnen Schritte in dem Unterpunkt „steps“ definiert. Diese Steps sind dann auch auf der Github Webseite ersichtlich sind.

Hier ein Bild, was für Inhalte sich gerade in meinem Repository befinden. Wie man erkennen kann, ist ein durch den Workflow entstandenes Package auch zu finden.

A screenshot of a computer screen

Description automatically generated with medium confidence

Und hier kann man den erfolgreich durchgelaufenen Workflow sehen in dem Actions Register.

Graphical user interface, application

Description automatically generated