



ESCOLA DE ENGENHARIA
Universidade Federal Fluminense



GitHub Actions

Tópicos Especiais em Sistemas de Energia Elétrica II: Desenvolvimento
Ágil com Python e Containers
Prof. Angelo Colombini

Rene Cruz Freire

Universidade Federal Fluminense
Campus Niterói

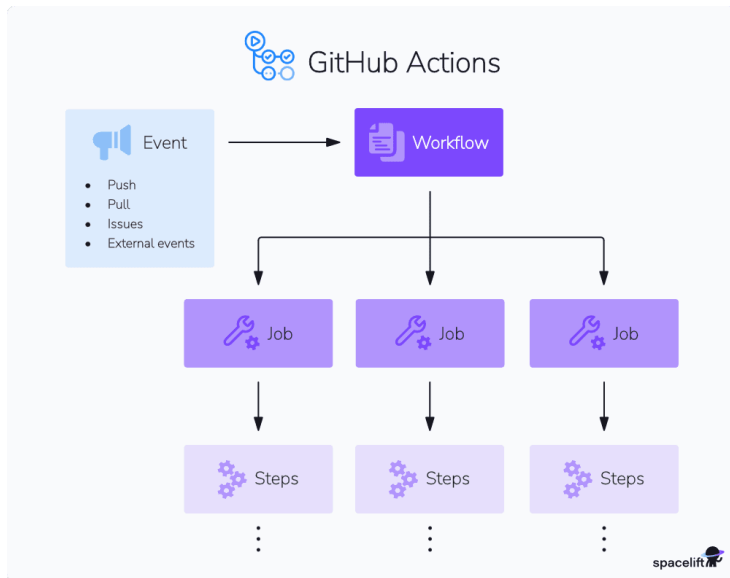
renefreire@id.uff.br

- 1 Visão Geral
- 2 Funcionamento
- 3 Curva de Aprendizado
- 4 Componentes
- 5 Vantagens e Desvantagens
- 6 Primeiro *Workflow* no GitHub Actions
- 7 Visualizando os Resultados do *Workflow*
- 8 Referências

- O GitHub Actions é uma plataforma de integração e entrega/deployment contínuos (CI/CD), que automatiza os *workflows* de desenvolvimento de software.
- Ele permite que você crie, teste e faça *deploy* de um código-fonte de software diretamente do seu repositório GitHub, criando fluxos de trabalho ou *pipelines* personalizados.
- Com várias opções de configuração para gatilhos baseados em *commits* e *merges*, o GitHub Actions é uma boa escolha para fluxos de trabalho baseados em GitOps.

- Os fluxos de trabalho do GitHub Actions são configurados usando arquivos YAML, que definem a sequência de tarefas ou ações a serem executadas quando acionadas por eventos como *pushes*, *pull requests* e *releases* de código.
- Ações são blocos reutilizáveis de código que executam tarefas específicas, como configurar dependências, executar testes e fazer o *deploy* em um provedor de nuvem ou servidores locais.
- É possível criar e publicar ações personalizadas para suas necessidades específicas.

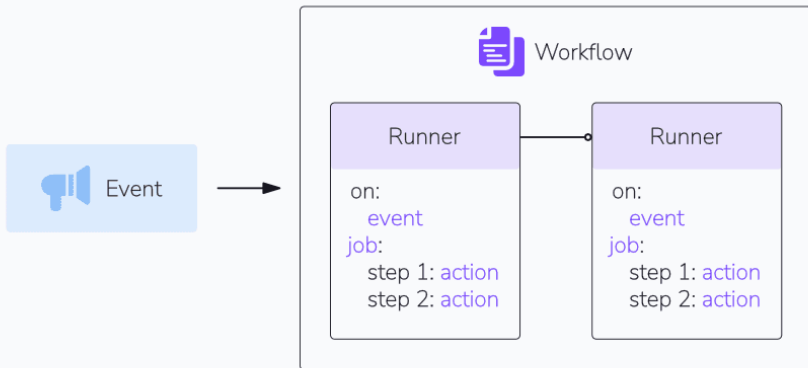
- O GitHub fornece máquinas virtuais, também conhecidas como *runners*, para executar seus fluxos de trabalho em ambientes Linux, Windows e macOS.
- Os *runners* oferecem suporte a qualquer linguagem de programação, plataforma e provedor de nuvem, tornando o GitHub flexível para vários projetos.
- O GitHub Actions integra-se perfeitamente com outros recursos do GitHub, como as *issues*, *pull requests* e *marketplace*, permitindo a criação de fluxos de trabalho automatizados com base em eventos no seu repositório.



- O GitHub Actions geralmente é considerado fácil de aprender, especialmente para desenvolvedores já familiarizados com o GitHub e conceitos básicos de *script*.
- Ele fornece um ponto de entrada fácil de usar para automação, permitindo que os desenvolvedores configurem rapidamente pipelines básicos de CI/CD e outros fluxos de trabalho sem amplo conhecimento de DevOps.
- Como acontece com qualquer ferramenta, dominar recursos mais avançados e casos de uso complexos exigirá mais tempo e prática.

- É possível configurar um fluxo de trabalho do GitHub Actions a ser disparado quando um evento ocorrer no seu repositório, como a abertura de uma solicitação de pull ou a criação de um problema.
- Um fluxo de trabalho contém um ou mais *jobs* que podem ser executados em ordem sequencial ou em paralelo.
- Cada *job* será executado em um *runner* próprio ou em um contêiner, e tem uma ou mais etapas que executam um *script* definido pelo usuário ou uma ação, que é uma extensão reutilizável que pode simplificar o fluxo de trabalho.

1. **Workflow:** é um processo automatizado definido por um arquivo YAML.
2. **Eventos:** são atividades específicas que acionam a execução de um fluxo de trabalho.
3. **Runners:** são máquinas virtuais que executam *jobs* em um fluxo de trabalho.
4. **Job:** consiste em uma série de etapas executadas, em paralelo ou sequencialmente, no mesmo *runner*.
5. **Steps:** são as tarefas ou comandos individuais que compõem um *job*.
6. **Ações:** são pacotes de código reutilizáveis usados como etapas em um fluxo de trabalho.



Vantagens

1. **Integração nativa com o GitHub:** Totalmente integrado ao ecossistema GitHub, o que facilita o uso e evita ferramentas externas para automação.
2. **Fácil de começar:** Com poucos cliques ou arquivos YAML simples, já é possível criar *workflows* de CI/CD rapidamente.
3. **Gratuito para projetos públicos:** Repositórios públicos têm acesso ilimitado aos *runners* hospedados pela plataforma.
4. **Flexibilidade de *workflows*:** Suporta múltiplos gatilhos (*push*, *pull request*, *cron*, *release*, etc.) e etapas condicionais, possibilitando pipelines altamente personalizados.
5. **Suporte a containers e ambientes personalizados:** Pode rodar *jobs* em containers Docker ou em VMs *self-hosted*, aumentando a flexibilidade do ambiente de *build*.

Desvantagens

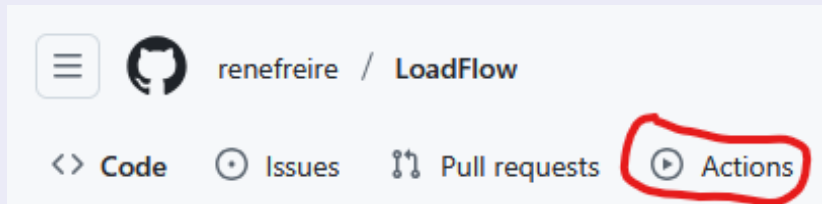
1. **Limitações no plano gratuito para repositórios privados:** Existe um limite de minutos e armazenamento para repositórios privados, especialmente em contas gratuitas.
2. **Performance e tempo de espera:** Em horários de pico ou em contas gratuitas, os runners hospedados podem ter filas ou performance inferior.
3. **Limitações nos logs:** Os logs podem ser truncados ou difíceis de depurar quando há muitos *steps* ou *jobs* em execução.
4. **Menos controle sobre infraestrutura dos runners hospedados:** Comparado a soluções como Jenkins em ambiente próprio, há menos controle sobre a configuração do ambiente.
5. **Falta de UI mais avançada para workflows complexos:** A interface do GitHub Actions ainda é relativamente simples para pipelines muito grandes ou com múltiplos caminhos condicionais.

Pré-requisitos

- É necessário um conhecimento básico do GitHub.
- Deve-se ter um repositório no GitHub para adicionar os arquivos.
- Deve-se ter acesso ao GitHub Actions.

Observações:

- Se a aba **Actions** não for exibida sob o nome do seu repositório, tal como na figura a seguir, provavelmente o Actions está desabilitado para o repositório.



1. No seu repositório no GitHub, crie um arquivo de *workflow* chamado `github-actions-demo.yml` no diretório `.github/workflows`.
 - ▶ Se o diretório `.github/workflows` já existir, navegue até ele no GitHub, clique em adicionar arquivo, depois clique em criar novo arquivo e nomeie o arquivo como `github-actions-demo.yml`.
 - ▶ Se o seu repositório não tiver um diretório `.github/workflows`, vá para a página principal do repositório no GitHub, clique em adicionar arquivo, depois clique em criar novo arquivo e nomeie o arquivo `.github/workflows/github-actions-demo.yml`.
 - ★ Isso cria os diretórios `.github` e `workflows` e o arquivo `github-actions-demo.yml` em uma única etapa.

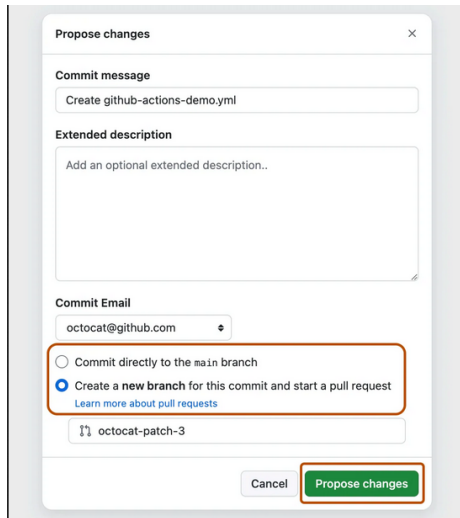
Observações:

- Para que o GitHub descubra quaisquer fluxos de trabalho do GitHub Actions no seu repositório, você deve salvar os arquivos de *workflow* em um diretório chamado `.github/workflows`.
- É possível dar ao arquivo de fluxo de trabalho o nome que desejar, mas deve-se usar `.yaml` ou `.yml` como extensão do nome do arquivo.
- YAML é uma linguagem de marcação comumente usada para arquivos de configuração.

2. Escreva o seguinte conteúdo no arquivo `.yaml`:

```
name: GitHub Actions Demo
run-name: ${ github.actor } is testing out GitHub Actions 🚀
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${ github.event_name } event."
      - run: echo "🏠 This job is now running on a ${ runner.os } server hosted by GitHub!"
      - run: echo "🔍 The name of your branch is ${ github.ref } and your repository is ${ github.repository }."
      - name: Check out repository code
        uses: actions/checkout@v4
      - run: echo "💡 The ${ github.repository } repository has been cloned to the runner."
      - run: echo "💻 The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${ github.workspace }
      - run: echo "🍏 This job's status is ${ job.status }."
```

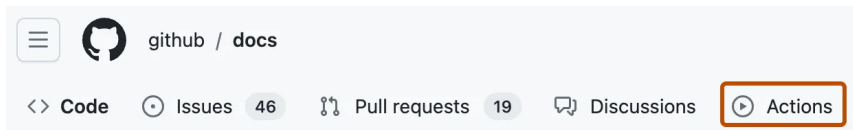

3. Clique em **commit changes**
4. Na caixa de diálogo **Propose changes**, selecione a opção de confirmar a *branch* padrão ou a opção de criar uma nova *branch* e iniciar um *pull request*, e em seguida clique em *commit changes* ou *propose changes*.



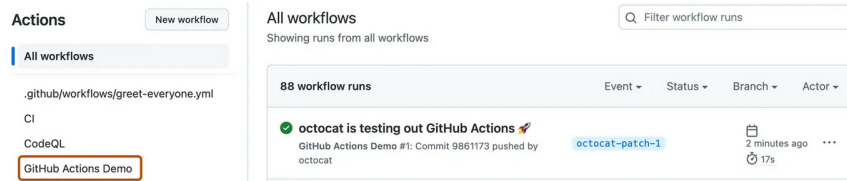
Observações:

- Enviar o arquivo de fluxo de trabalho para uma ramificação no seu repositório aciona o evento push e executa seu fluxo de trabalho.
- Se você optar por iniciar uma solicitação de pull, poderá continuar e criá-la, mas isso não é necessário para os propósitos deste início rápido porque a confirmação ainda foi feita em uma ramificação e acionará o novo fluxo de trabalho.

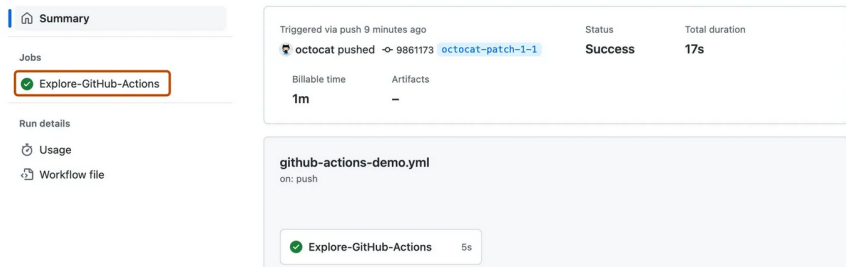
1. No GitHub, navegue até a página principal do repositório.
2. Abaixo do nome do repositório, clique em **Actions**.



3. Na barra lateral esquerda, clique no fluxo de trabalho que deseja exibir, neste exemplo "GitHub Actions Demo".



- Na lista de execuções de fluxo de trabalho, clique no nome da execução que deseja ver. Neste exemplo, "USERNAME is testing out GitHub Actions".
- Na barra lateral esquerda da página de execução do *workflow*, em Jobs, clique em Explore-GitHub-Actions.



Summary

Jobs

- ✓ Explore-GitHub-Actions

Run details

- 🕒 Usage
- 📄 Workflow file

Triggered via push 9 minutes ago

octocat pushed → 9861173 octocat-patch-1-1

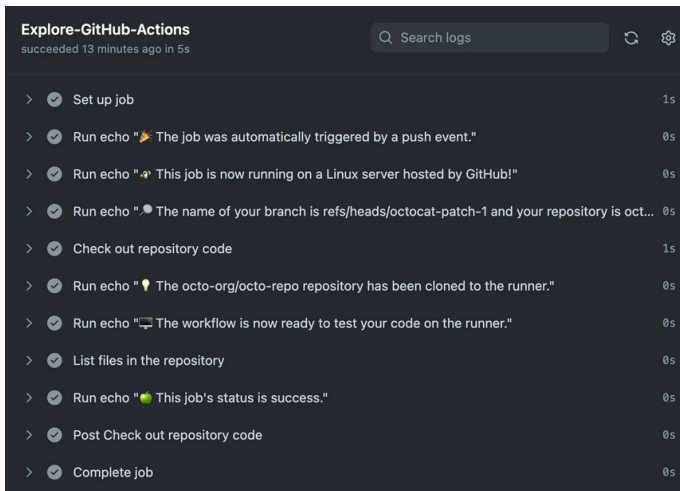
Status: **Success** Total duration: **17s**

Billable time: **1m** Artifacts: —

github-actions-demo.yml
on: push

- ✓ Explore-GitHub-Actions 5s

6. O log mostra como cada etapa foi processada. Expanda qualquer uma das etapas para ver seus detalhes.



Explore-GitHub-Actions
succeeded 13 minutes ago in 5s

Search logs

- > ✓ Set up job 1s
- > ✓ Run echo "👉 The job was automatically triggered by a push event." 0s
- > ✓ Run echo "👉 This job is now running on a Linux server hosted by GitHub!" 0s
- > ✓ Run echo "🗨️ The name of your branch is refs/heads/octocat-patch-1 and your repository is oct..." 0s
- > ✓ Check out repository code 1s
- > ✓ Run echo "💡 The octo-org/octo-repo repository has been cloned to the runner." 0s
- > ✓ Run echo "💻 The workflow is now ready to test your code on the runner." 0s
- > ✓ List files in the repository 0s
- > ✓ Run echo "🟢 This job's status is success." 0s
- > ✓ Post Check out repository code 0s
- > ✓ Complete job 0s

- <https://docs.github.com/en/actions/writing-workflows/quickstart> - Inicializando um *workflow* no *GitHub Actions*.
- <https://docs.github.com/en/actions/about-github-actions/understanding-github-actions> - Visão geral sobre o *GitHub Actions*.
- <https://docs.github.com/en/actions/writing-workflows/about-workflows#understanding-the-workflow-file> - Aprofundamento sobre os *workflows* no *GitHub Actions*
- <https://docs.github.com/en/enterprise-cloud@latest/actions/guides> - Listagem de guias sobre o *GitHub Actions*
- <https://spacelift.io/blog/github-actions-tutorial> - Tutorial sobre o *GitHub Actions*