



Introdução ao **GitHub Actions**





Introdução ao GitHub Actions

Neste módulo, você aprenderá o que são o GitHub Actions, o fluxo de ação e seus elementos. Saiba o que são eventos, explore trabalhos e executores e como ler a saída de ações do console.

Objetivos de aprendizagem

Ao final deste módulo, você será capaz de fazer o seguinte:

- Explicar GitHub Actions e fluxos de trabalho
- Criar e trabalhar com o GitHub Actions e fluxos de trabalho
- Descrever eventos, trabalhos e executores
- Examinar o gerenciamento de saída e de versão para ações

Pré-requisitos

Nenhum

Este módulo faz parte destes roteiros de aprendizagem

- [AZ-400: Implementar a CI com o Azure Pipelines e o GitHub Actions](#)
- [Explore o Azure DevOps para simplificar seu processo de desenvolvimento](#)
- [Gerenciamento do ciclo de vida do aplicativo para o Power Platform](#)

Introdução

O que são as ações?

Explorar o fluxo de Actions

Noções básicas de fluxos de trabalho

Descrever elementos de sintaxe de fluxo de trabalho padrão

Explorar Eventos

Explorar trabalhos

Explorar os executores

Examinar a versão e testar uma ação

Resumo





Introdução

As GitHub Actions são o mecanismo principal para automação no GitHub.

Elas podem ser usadas para uma ampla variedade de finalidades, mas um dos mais comuns é implementar a integração contínua.

Neste módulo, você aprenderá o que são o GitHub Actions, o fluxo de ação e seus elementos. Entender o que são eventos, explorar trabalhos e executores e como ler a saída do console de ações.

Objetivos de aprendizagem

Depois de concluir este módulo, **os alunos e profissionais poderão:**

- Explicar GitHub Actions e fluxos de trabalho.
- Criar e trabalhar com o GitHub Actions e fluxos de trabalho.
- Descrever eventos, trabalhos e executores.
- Examine a saída e o gerenciamento de versão para ver se há ações.

Pré-requisitos

- Noções básicas sobre o que é DevOps e seus conceitos.
- É útil estar familiarizado com os princípios de controle de versão, mas não é obrigatório.
- É benéfico ter experiência em uma organização que fornece software.

O que são as ações?

As ações são o mecanismo usado para fornecer automação de fluxo de trabalho dentro do ambiente do GitHub.

Geralmente, são usadas para criar soluções de CI (integração contínua) e CD (implantação contínua).

No entanto, **podem ser usadas para uma ampla variedade de tarefas:**

- Teste automatizado.
- Responder automaticamente a novos problemas, menções.
- Disparar revisões de código.
- Tratar solicitações de pull.
- Gerenciamento de branch.

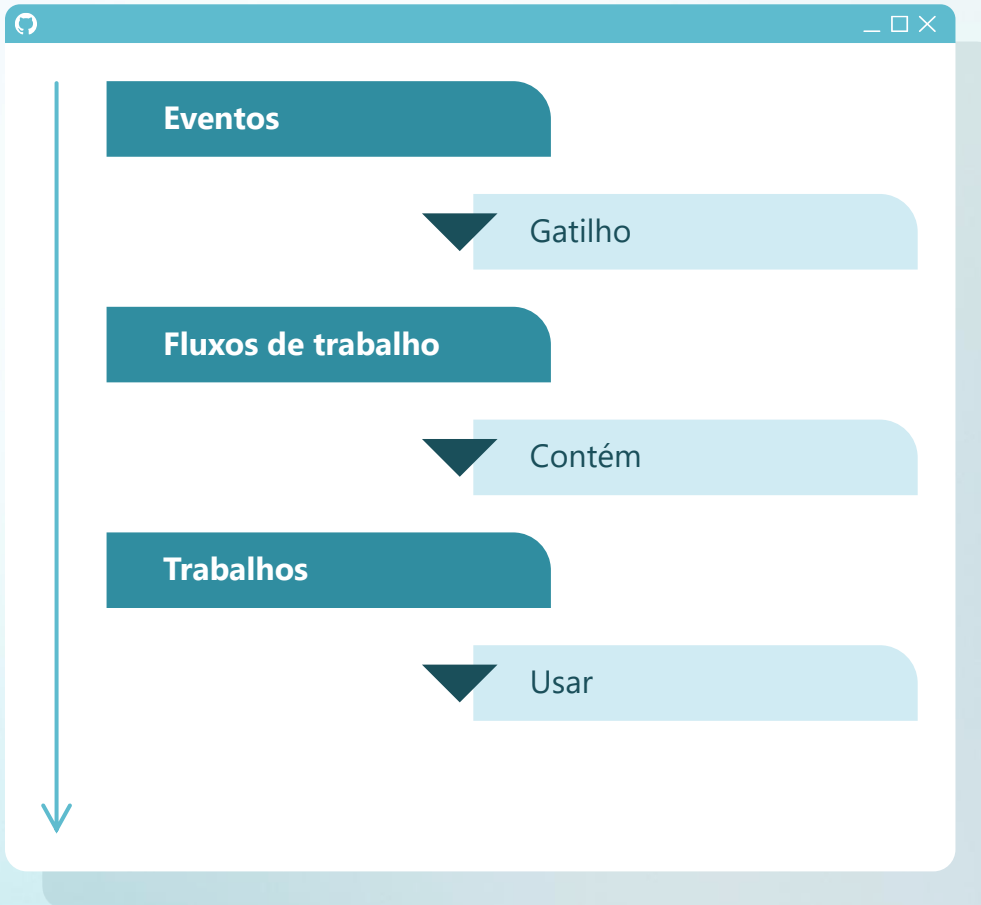
Elas são definidos no YAML e permanecem dentro de repositórios do GitHub.

As ações são executadas em "executores", sejam hospedados pelo GitHub ou auto-hospedados. As ações de contribuição podem ser encontradas no GitHub Marketplace. Confira [Ações do Marketplace](#).





Explorar o fluxo de Action



O GitHub rastreia eventos que ocorrem.

Os eventos podem disparar o início dos fluxos de trabalho.

Os fluxos de trabalho também podem iniciar conforme cronogramas baseados em cron e ser disparados por eventos fora do GitHub.

Eles podem ser disparados manualmente.

Fluxos de trabalho são a unidade de automação. Eles contêm Trabalhos.

Os trabalhos usam Ações para fazer o trabalho.

Explorar o fluxo de Action

Os fluxos de trabalho definem a automação necessária.

Ele detalha os eventos que devem disparar o fluxo de trabalho.

Além disso, defina os trabalhos que devem ser executados quando o fluxo de trabalho é disparado.

O trabalho define o local no qual as ações serão executadas, por exemplo, qual executor usar.





Os fluxos de trabalho são escritos em YAML e ao vivo em um repositório do GitHub no local **.github/workflows**.

Fluxo de trabalho de exemplo:

```
YAML Copiar

# .github/workflows/build.yml
name: Node Build.

on: [push]

jobs:
  mainbuild:

    runs-on: ${ matrix.os }

    strategy:
      matrix:
        node-version: [12.x]
        os: [windows-latest]

    steps:
      - uses: actions/checkout@v1
      - name: Run node.js on latest Windows.
        uses: actions/setup-node@v1
        with:
          node-version: ${ matrix.node-version }
      - name: Install NPM and build.
        run: |
          npm ci
          npm run build
```

Você pode encontrar um conjunto de fluxos de trabalho de início aqui:

Fluxos de trabalho iniciais.

Você pode ver aqui a sintaxe acessível para fluxos de trabalho:

sintaxe de fluxo de trabalho para o GitHub Actions.

Descrever elementos de sintaxe de fluxo de trabalho padrão

Os fluxos de trabalho incluem vários elementos de sintaxe padrão.

- **Nome:** é o nome do fluxo de trabalho. É opcional, mas altamente recomendável. Aparece em vários locais na interface do usuário do GitHub.
- **On:** é o evento ou a lista de eventos que dispararão o fluxo de trabalho.
- **Trabalhos:** é a lista de trabalhos a serem executados. Os fluxos de trabalho podem conter um ou mais trabalhos.
- **Executa em:** informa a Actions que executor usar.





- **Etapas:** é a lista de etapas para o trabalho. As etapas dentro de um trabalho são executadas no mesmo executor.
- **Usa:** informa a Actions que ação predefinida precisa ser recuperada. Por exemplo, você pode ter uma ação que instala o Node.js.
- **Executar:** informa o trabalho para executar um comando no executor. Por exemplo, você pode executar um comando NPM.

Você pode ver aqui a sintaxe acessível para fluxos de trabalho: [sintaxe de fluxo de trabalho para o GitHub Actions](#).

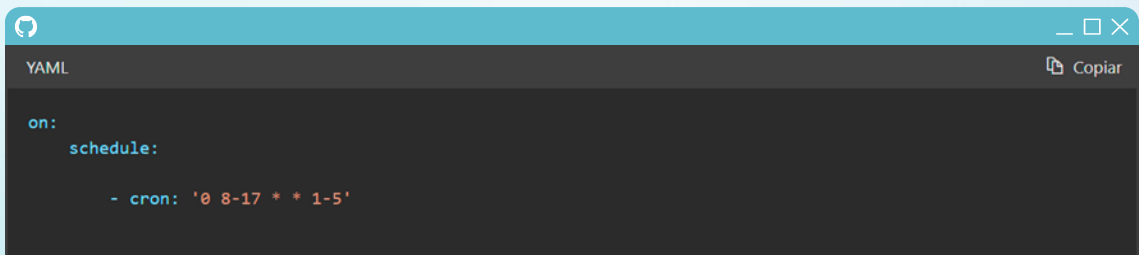
Explorar Eventos

Os eventos são implementados pela cláusula `on` em uma definição de fluxo de trabalho.

Há vários tipos de eventos que podem disparar fluxos de trabalho.

Eventos agendados

Com esse tipo de gatilho, um agendamento cron precisa ser fornecido.



```
YAML
on:
  schedule:
    - cron: '0 8-17 * * 1-5'
```

Os agendamentos do Cron são **baseados em cinco valores**:

- Minuto (0 a 59)
- Hora (0 a 23)
- O dia do mês (1 a 31)
- Mês (1 a 12)
- Dia da semana (0 a 6)

Aliases para os meses são JAN-DEC e, para os dias da semana, são SUN-SAT.

Um curinga significa qualquer um. (* é um valor especial em YAML, portanto, a cadeia de caracteres cron precisará estar entre aspas)

Portanto, no exemplo acima, o agendamento seria das 8h às 17h de segunda a sexta-feira.



Eventos de código

Eventos de código dispararão a maioria das ações. Isso acontece quando ocorre um evento de interesse no repositório.

```
YAML Copiar

on:
  pull_request
```

O evento acima seria disparado quando ocorresse uma solicitação de pull.

```
YAML Copiar

on:
  [push, pull_request]
```

O evento acima seria disparado quando ocorresse uma solicitação de pull ou push.

```
YAML Copiar

on:
  pull_request:
    branches:
      - develop
```

O evento mostra como ser específico sobre a seção relevante do código.

Nesse caso, será disparado quando uma solicitação de pull for feita na ramificação de desenvolvimento.

Eventos manuais

Há um evento exclusivo usado para disparar as execuções de fluxo de trabalho manualmente. Use o evento `workflow_dispatch`.

Seu fluxo de trabalho deve estar na ramificação padrão do repositório.



Eventos do webhook

Fluxos de trabalho podem ser executados quando um webhook do GitHub é chamado.

```
YAML
on:
  gollum
```

Esse evento seria disparado quando alguém atualizasse (ou criasse primeiro) uma página do Wiki.

Eventos externos

Os eventos podem estar em **repository_dispatch**. Isso permite que os eventos sejam disparados de sistemas externos.

Para obter mais informações sobre eventos, confira [Eventos que disparam fluxos de trabalho](#).

Explorar trabalhos

Os fluxos de trabalho contêm um ou mais trabalhos. Um trabalho é um conjunto de etapas que serão executadas em ordem em um executor.

As etapas dentro de um trabalho são executadas no mesmo executor e compartilham o mesmo sistema de arquivos.

Os logs produzidos por trabalhos são pesquisáveis e os artefatos produzidos podem ser salvos.

Trabalhos com dependências

Por padrão, se um fluxo de trabalho contiver vários trabalhos, eles serão executados em paralelo.

```
YAML
jobs:
  startup:
    runs-on: ubuntu-latest
    steps:
      - run: ./setup_server_configuration.sh
  build:
    steps:
      - run: ./build_new_server.sh
```





Às vezes, talvez seja necessário que um trabalho aguarde a conclusão de outro. Você pode fazer isso definindo dependências entre os trabalhos.

```
YAML                                                                    Copiar

jobs:
  startup:
    runs-on: ubuntu-latest
    steps:

      - run: ./setup_server_configuration.sh
  build:
    needs: startup
    steps:

      - run: ./build_new_server.sh
```

Observação

Se o trabalho de inicialização no exemplo acima falhar, o trabalho de build não será executado.

Para obter mais informações sobre dependências de trabalho, confira a seção **Como criar trabalhos dependentes** em **Como gerenciar fluxos de trabalho complexos**

Explorar os executores

Os executores do GitHub são recursos de computação que executam fluxos de trabalho do GitHub Actions. Cada executor pode executar apenas um trabalho de cada vez. Eles permitem que os desenvolvedores executem tarefas de build, de teste e de implantação diretamente nos repositórios do GitHub.

Há dois tipos principais de executores do GitHub:

- Os executores hospedados no GitHub são recursos de computação virtualizados ou em contêineres, fornecidos e gerenciados pelo GitHub.
- Os executores auto-hospedados são recursos de computação físicos, virtualizados ou em contêineres que os usuários e organizações do GitHub provisionam e gerenciam por conta própria.

Cada tipo tem algumas características exclusivas, apresenta uma série de funcionalidades distintas e garante várias considerações diferentes.

É importante observar que o GitHub recomenda fortemente o uso de executores auto-hospedados em repositórios públicos. Isso introduz um risco de segurança significativo, pois potencialmente permite que qualquer pessoa execute código dentro do ambiente privado da organização.





Executores hospedados no GitHub

Os executores hospedados no GitHub **oferecem uma solução conveniente para executar fluxos de trabalho no GitHub Actions**, eliminando a necessidade de administrar os componentes de hardware e software subjacentes. Eles são projetados para serem dimensionados automaticamente com base na demanda, garantindo um desempenho ideal durante os períodos de pico de uso.

O GitHub fornece vários ambientes pré-configurados para executores hospedados no GitHub, abrangendo diferentes configurações de software e sistemas operacionais, incluindo Ubuntu Linux, Microsoft Windows e macOS.

Os executores hospedados no GitHub incluem as ferramentas internas padrão do sistema operacional. Por exemplo, os executores do Ubuntu e do macOS incluem `grep`, `find` e `which`. Para identificar todas as outras ferramentas pré-instaladas em executores, os usuários podem examinar a SBOM (lista de materiais de software) para cada build das imagens do executor do Windows e do Ubuntu.

Como alternativa, os usuários podem examinar a subseção Imagem do Executor da seção Configurar trabalho nos logs de fluxo de trabalho.

O link após a entrada Software Incluído descreve todas as ferramentas pré-instaladas no executor que executou o fluxo de trabalho. Também é possível instalar software adicional em executores hospedados no GitHub criando um trabalho que instala os pacotes como parte do fluxo de trabalho existente.

Os executores hospedados no GitHub são executados na infraestrutura de nuvem do GitHub, aproveitando máquinas virtuais ou contêineres para executar fluxos de trabalho. Cada execução de fluxo de trabalho é isolada em um ambiente próprio, garantindo segurança e reprodutibilidade. Os executores hospedados no GitHub integram-se perfeitamente ao GitHub Actions, permitindo que os usuários os referenciem diretamente nos fluxos de trabalho hospedados nos repositórios do GitHub.

Há alguns limites para o uso do GitHub Actions ao usar os executores hospedados no GitHub. Em particular, cada trabalho em um fluxo de trabalho tem o máximo de seis horas de tempo de execução. Se um trabalho atingir esse limite, o trabalho será terminado e não será completado. Cada execução de fluxo de trabalho é limitada a 35 dias.

Se uma execução de fluxo de trabalho atingir esse limite, a execução dela será cancelada. Esse período inclui a duração da execução e o tempo gasto em espera e aprovação.

Pré-requisitos

Antes de implementar os executores hospedados no GitHub, os usuários precisam ter um repositório do GitHub em que possam definir fluxos de trabalho usando o GitHub Actions. Os executores estão disponíveis para todos os usuários do GitHub com acesso ao GitHub Actions.





Implementação

Ao contrário dos executores auto-hospedados, aqueles hospedados no GitHub são provisionados automaticamente como parte de uma execução de fluxo de trabalho individual. Os usuários definem fluxos de trabalho como arquivos formatados em YAML armazenados no diretório

.github/workflows em repositórios do GitHub. Dentro da configuração do fluxo de trabalho, os usuários especificam o ambiente desejado do executor, incluindo o sistema operacional e as dependências de software. Os executores com especificações correspondentes são configurados sob demanda sempre que o fluxo de trabalho é disparado, com um executor por trabalho. Os gatilhos podem ser manuais ou automáticos, com base em eventos como pushes de código, solicitações de pull ou eventos de expedição de repositório. Os executores hospedados no GitHub autenticam-se com o GitHub usando tokens ou credenciais fornecidas pelo GitHub Actions. Eles dependem da conectividade interna para se comunicar com a plataforma GitHub e baixar artefatos de fluxo de trabalho.

Manutenção

O GitHub gerencia atualizações e manutenção de executores hospedados no GitHub, garantindo que eles permaneçam atualizados com as versões de software mais recentes e patches de segurança. As ferramentas de software incluídas nos executores são atualizadas semanalmente. As atividades do executor são monitoradas e registradas em log, facilitando o acompanhamento de execuções de fluxo de trabalho e solução de problemas.

Licenciamento e custo

Os executores hospedados no GitHub estão incluídos nos preços do GitHub Actions, com cobrança baseada em uso para minutos de fluxo de trabalho além da camada gratuita. Os usuários se beneficiam da escala automatizada e econômica, pois o GitHub provisiona e desaloca automaticamente os executores com base na demanda.

Executores auto-hospedados

Em comparação com os executores hospedados no GitHub, os auto-hospedados fornecem opções de controle e personalização maiores, com ambientes de execução capazes de acomodar uma gama mais ampla de requisitos. Eles podem ser implantados localmente ou na nuvem, dependendo de critérios como conectividade de rede, custo e disponibilidade de recursos.

Os executores auto-hospedados são provisionados e gerenciados pelos usuários, dando-lhes controle total sobre o ambiente de execução. Eles são totalmente personalizáveis, incluindo especificações de hardware, configurações de software e configurações de rede. Eles também facilitam a integração com a infraestrutura e as ferramentas existentes, minimizando a possibilidade de problemas de compatibilidade e interoperabilidade.





Ao contrário dos executores hospedados pelo GitHub, não há limites para o tempo necessário para concluir uma execução de trabalho individual e execuções de fluxo de trabalho.

Pré-requisitos

Os usuários precisam preparar e configurar executores auto-hospedados na infraestrutura escolhida por eles, incluindo a instalação do software executor e quaisquer dependências de software adicionais. O código-fonte para os executores auto-hospedados está disponível como um projeto de código aberto no GitHub em <https://github.com/actions/runner>. Cada executor auto-hospedado atua como um agente que se comunica com o GitHub Actions para executar fluxos de trabalho.

Os executores auto-hospedados exigem conectividade de rede de saída, credenciais de autenticação e autorização para acessar a plataforma GitHub e baixar artefatos de fluxo de trabalho. Dependendo do local dos executores, talvez seja necessário configurar regras de firewall para atender a esses requisitos.

Implementação

Assim como acontece com os executores hospedados no GitHub, a implementação envolve a definição de fluxos de trabalho formatados por YAML e o armazenamento do diretório `.github/workflows` nos repositórios do GitHub.

No entanto, para que os fluxos de trabalho usem executores auto-hospedados, os usuários precisam registrá-los primeiro, fornecendo os tokens ou credenciais de autenticação necessários. Como parte do registro, os usuários especificam características como o nome do executor, bem como rótulos e parâmetros do ambiente de execução.

O registro pode ocorrer em diferentes níveis em uma empresa:

- Nível do repositório (repositório único)
- Nível organizacional (vários repositórios em uma organização)
- Nível empresarial (várias organizações em uma empresa)

Manutenção

Os usuários são responsáveis por atualizar e manter executores auto-hospedados, incluindo a instalação de atualizações de software e patches de segurança. A manutenção também envolve o monitoramento da integridade e do desempenho do executor, bem como a solução de problemas que surgem em todo o runtime do executor.

Licenciamento e custo

Os executores auto-hospedados não incorrem em encargos de licenciamento adicionais além dos preços do GitHub Actions e dos custos de infraestrutura associados, incluindo computação, armazenamento e rede. Otimizar a alocação e a utilização de recursos torna-se responsabilidade do usuário.





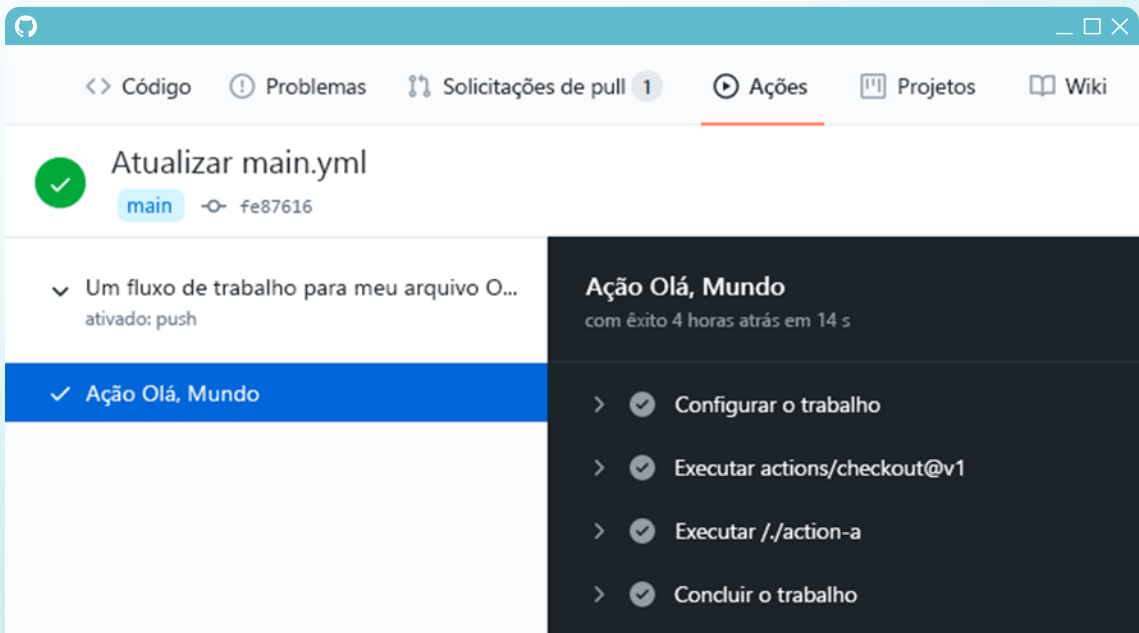
Examinar a versão e testar uma ação

As ações geralmente produzirão a saída do console. Você não precisa se conectar diretamente aos executores para recuperar essa saída.

A saída do console das ações está disponível diretamente de dentro da interface do usuário do GitHub.

Selecione **Ações** no menu do repositório superior para ver uma lista de fluxos de trabalho executados para ver a saída.

Em seguida, clique no nome do trabalho para ver a saída das etapas.



A saída do console pode ajudar na depuração.

Se não for suficiente, você também poderá habilitar mais registro em log.

Confira: **[Como habilitar o registro em log da depuração.](#)**

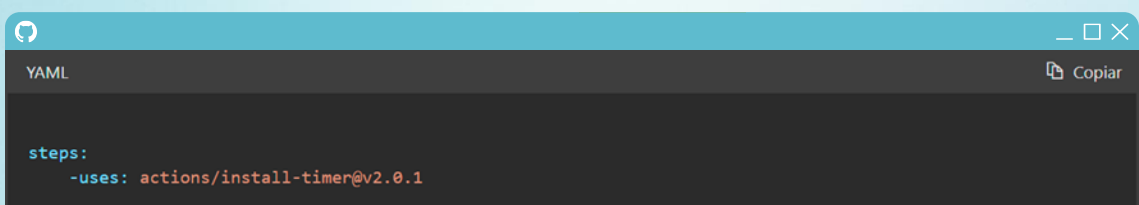
Release Management para Actions

Embora você possa estar feliz em recuperar a versão mais recente da ação, há muitas situações em que você pode querer uma versão específica da ação.

Você pode solicitar uma versão específica da ação de várias maneiras:

Marcações

As tags permitem que você especifique as versões precisas com que deseja trabalhar.





Hashes baseados em SHA

Você pode especificar um hash baseado em SHA solicitado para uma ação.

Ele garante que a ação não foi alterada. No entanto, a desvantagem disso é que você também não receberá atualizações para a ação automaticamente.

```
YAML Copiar
steps:
  -uses: actions/install-timer@327239021f7cc39fe7327647b213799853a9eb98
```

Branches

Uma forma comum de solicitar ações é se referir à ramificação com que você quer trabalhar. Em seguida, você obterá a versão mais recente dessa ramificação. Isso significa que você se beneficiará das atualizações, mas também aumentará a chance de interrupção do código.

```
YAML Copiar
steps:
  -uses: actions/install-timer@develop
```

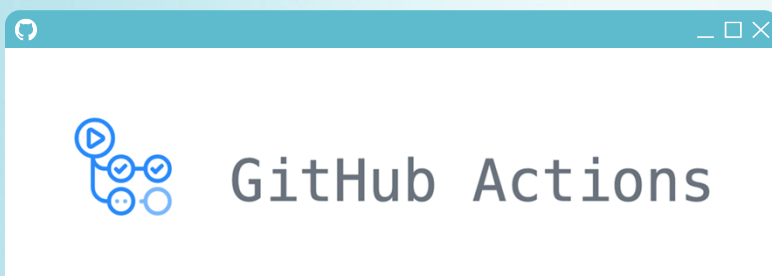
Testar uma ação

O GitHub oferece várias ferramentas de aprendizado para ações.

GitHub Actions: **hello-world**

Você verá um exemplo básico de como:

- Organizar e identificar arquivos de fluxo de trabalho.
- Adicionar scripts executáveis.
- Criar blocos de ação e fluxo de trabalho.
- Disparar fluxos de trabalho.
- Descobrir logs de fluxo de trabalho.





Resumo

Neste módulo, você aprendeu o que GitHub Actions, o fluxo de ação e seus elementos. Além disso, entendeu o que são eventos, trabalhos explorados e executores e como ler a saída do console de ações.

Você aprendeu a descrever os benefícios e o uso de:

- Explicar GitHub Actions e fluxos de trabalho.
- Criar e trabalhar com o GitHub Actions e fluxos de trabalho.
- Descrever eventos, trabalhos e executores.
- Examine a saída e o gerenciamento de versão para ver se há ações.

Saiba mais

- [Início Rápido para GitHub Actions.](#)
- [Sintaxe de fluxo de trabalho para GitHub Actions – GitHub Docs.](#)
- [Eventos que disparam fluxos de trabalho – GitHub Docs.](#)

Para te ajudar a começar a usar as ferramentas que acelerarão a sua produção com menos esforço de desenvolvimento, disponibilizamos os links para acessá-las:

GitHub Copilot: <https://aka.ms/githubcopilotquickstart>

GitHub Codespaces: <https://aka.ms/githubcodespacesquickstart>

GitHub Actions: <https://aka.ms/githubactionsquickstart>

