

Tutorial - Como instalar o Argo Rollouts

Índice

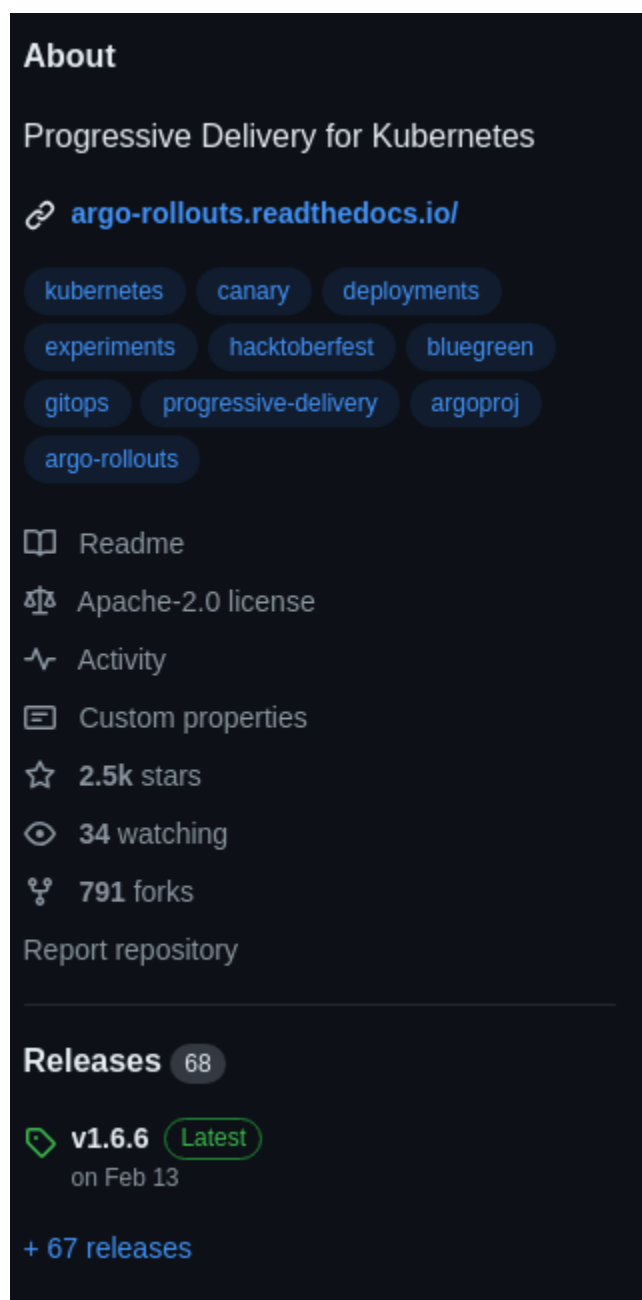
- [Introdução](#)
- [Instalação do Controller e Dashboard](#)
- [Instalação do plugin do kubectl](#)
 - [Linux](#)
 - [Windows](#)

Introdução

Nesta página será descrita as etapas necessárias para a instalação do Argo Rollouts (Controller e Dashboard) bem como o plugin que estende os comandos do kubectl.

O Argo Rollouts é um projeto Open Source que está disponível no repositório [argo-rollouts do projeto argoproj](#) no Github.

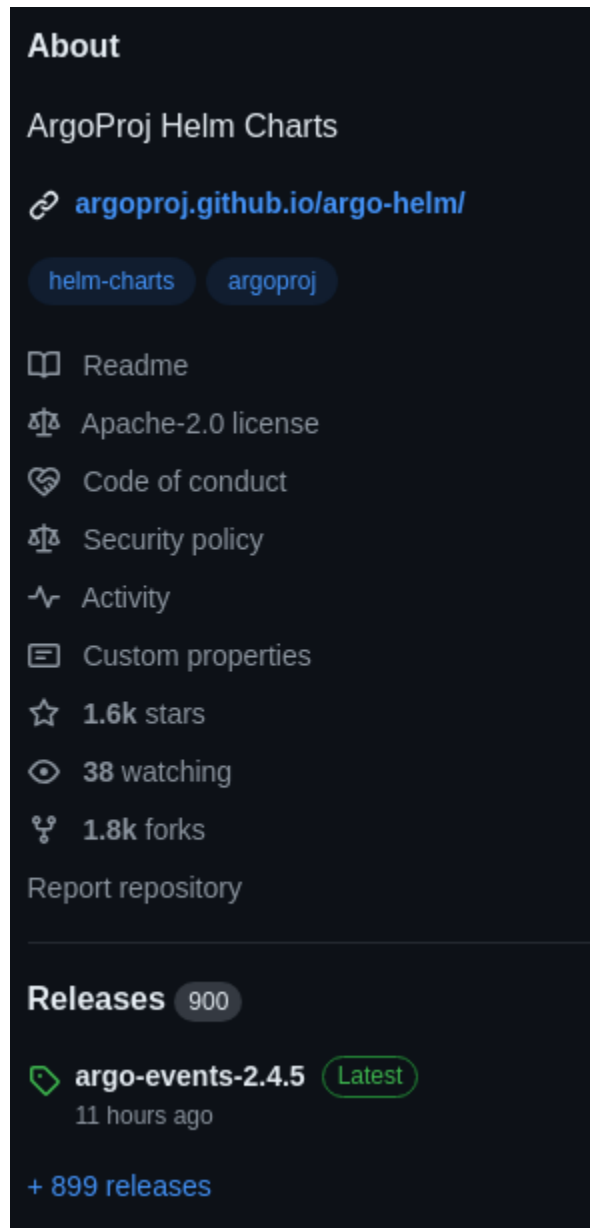
Os releases da aplicação podem ser consultados através do menu da lateral direita > Releases:



The screenshot shows the GitHub repository page for Argo Rollouts. The page has a dark theme. At the top, it says 'About' and 'Progressive Delivery for Kubernetes'. Below that is a link to 'argo-rollouts.readthedocs.io/'. There are several topic tags: 'kubernetes', 'canary', 'deployments', 'experiments', 'hacktoberfest', 'bluegreen', 'gitops', 'progressive-delivery', 'argoproj', and 'argo-rollouts'. On the left sidebar, there are links to 'Readme', 'Apache-2.0 license', 'Activity', 'Custom properties', '2.5k stars', '34 watching', and '791 forks'. At the bottom, there is a 'Releases' section with a count of 68 releases. The latest release is 'v1.6.6' on Feb 13, marked as 'Latest'. There is a link to '+ 67 releases'.

O projeto também disponibiliza um Helm Chart que está disponível no repositório [argo-helm](https://github.com/argoproj/argo-helm).

Os releases do Helm Chart podem ser consultados através do menu da lateral direita > Releases:



About

ArgoProj Helm Charts

argoproj.github.io/argo-helm/

helm-charts argoproj

Readme

Apache-2.0 license

Code of conduct

Security policy

Activity

Custom properties

1.6k stars

38 watching

1.8k forks

Report repository

Releases 900

argo-events-2.4.5 Latest

11 hours ago

+ 899 releases

Na lista de releases procure por argo-rollouts:

Mar 26

github-actions

argo-rollouts-...

38900af

Compare

argo-rollouts-2.35.1

A Helm chart for Argo Rollouts

What's Changed

- chore(deps): update renovatebot/github-action action to v40.1.6 by @argoproj-renovate in [#2602](#)
- chore(deps): update actions/create-github-app-token action to v1.9.1 by @argoproj-renovate in [#2604](#)
- feat(argo-rollouts): add minimum RBAC for Gateway API by @congiv in [#2599](#)

New Contributors

- @congiv made their first contribution in [#2599](#)

Full Changelog: [argo-workflows-0.41.0...argo-rollouts-2.35.1](#)

Contributors

congiv

▼ Assets 4

argo-rollouts-2.35.1.tgz	53 KB	Mar 26
argo-rollouts-2.35.1.tgz.prov	1.59 KB	Mar 26
Source code (zip)		Mar 26
Source code (tar.gz)		Mar 26

Instalação do Controller e Dashboard



Automação

Será criado uma automação no Cockpit para realizar a instalação / atualização do Argo Rollouts

Primeiramente verifique se o seu kubectl está apontando para o ambiente desejado através do comando:

```
kubectl config get-contexts
```

Identifique qual contexto está atualmente ativo. Para isso, na coluna current deve existir um *:

```
[c99303a@VDICTXFDRH8151 ~]$ kubectl config get-contexts
CURRENT  NAME          CLUSTER                                AUTHINFO                                NAMESPACE
*         br.prod       digital-prod.sa-east-1.eksctl.io       kubernetes-dashboard@digital-prod.sa-east-1.eksctl.io
         br.sandbox    arn:aws:eks:sa-east-1:038004596529:cluster/digital-sandbox
         br.uat        digital-uat.sa-east-1.eksctl.io        kubernetes-dashboard@digital-uat.sa-east-1.eksctl.io
         minikube      minikube                               minikube                                default
         us.dev        digital-dev.us-east-1.eksctl.io        kubernetes-dashboard@digital-dev.us-east-1.eksctl.io
```

Neste exemplo, o contexto atual é us.dev. Para alterar o contexto execute o comando:

```
kubectl config use-context <nome do contexto>
```

Após mudar o contexto será apresentado uma mensagem confirmando a mudança:

```
[c99303a@VDICTXFDRH8151 ~]$ kubectl config use-context br.uat  
Switched to context "br.uat".
```

Execute o seguinte comando para criar um novo namespace:

```
kubectl create namespace deployment-system
```

Execute o seguinte comando para injetar a subida do container istio-proxy em todos os pods do namespace:

```
kubectl label namespace deployment-system istio-injection=enabled --overwrite
```

Em Digital PaaS armazenamos o Helm Chart do Argo Rollouts na pasta argo-rollouts/chart no repositório [digital-monitoring](#). Nesta pasta contém um release específico do Helm Chart oficial do Argo Rollouts sem qualquer modificação.

O motivo é queremos ter total controle sobre os arquivos do Helm Chart a serem utilizados.

O values.yaml utilizado durante a instalação da ferramenta encontra-se na pasta argo-rollouts/configs/chart/general.

As seguintes alterações foram realizadas no values.yaml:

```
controller:  
  # -- Rodar os pods apenas nas máquinas de infra  
  nodeSelector:  
    Worker: infra  
  tolerations:  
    - key: dedicated  
      operator: Equal  
      value: infra  
      effect: NoSchedule  
  # -- Rodar os pods em máquina diferentes de maneira preferencial  
  affinity:  
    podAntiAffinity:  
      preferredDuringSchedulingIgnoredDuringExecution:  
        - weight: 100  
          podAffinityTerm:  
            labelSelector:  
              matchExpressions:  
                - key: app.kubernetes.io/component  
                  operator: In  
                  values:  
                    - rollouts-controller  
            topologyKey: kubernetes.io/hostname  
  logging:  
    # -- Alterar o log para o formato json para facilitar futuros parses  
    format: "json"  
  # -- Definir a quantidade inicial e os limites de recursos a serem consumidos pelo controller  
  resources:  
    limits:  
      cpu: 300m  
      memory: 512Mi  
    requests:  
      cpu: 50m  
      memory: 64Mi  
  metrics:  
    # -- Habilitar o service para obtermos as métricas do controller  
    enabled: true  
    serviceMonitor:  
      # -- Habilitar o ServiceMonitor para coleta das métricas pelo Prometheus
```

```

    enabled: true
    # -- Adicionar as labels necessárias para que o ServiceMonitor seja processado pelo Prometheus
    additionalLabels:
      release: kube-prometheus-stack
providerRBAC:
  # -- Desabilitar os RBACs desnecessários
  providers:
    smi: false
    ambassador: false
    awsLoadBalancerController: false
    awsAppMesh: false
    traefik: false
    apisix: false
    contour: false
    glooPlatform: false
    gatewayAPI: false
dashboard:
  # -- Habilitar o deploy do dashboard server
  enabled: true
  # -- Definir cluster role como readonly (Ficará como somente leitura até que seja possível limitar as
  # pessoas que podem executar as ações)
  readonly: true
  # -- Rodar apenas nas máquinas de infra
  nodeSelector:
    Worker: infra
  tolerations:
    - key: dedicated
      operator: Equal
      value: infra
      effect: NoSchedule
  # -- Definir a quantidade inicial e os limites de recursos a serem consumidos pelo controller
  resources:
    limits:
      memory: 512Mi
      cpu: 500m
    requests:
      memory: 32Mi
      cpu: 10m

```

Baixe o repositório [digital-monitoring](#) e acesse o diretório argo-rollouts pelo terminal.

Execute o seguinte comando para instalar/atualizar o Argo Rollouts:

```
helm upgrade --install argo-rollouts chart/ --namespace=deployment-system --values=configs/chart/general/values.yaml
```

Caso as outras BUs não queiram utilizar o nosso repositório, pode-se utilizar o seguinte comando para adicionar o repositório oficial de Helm Charts do Argo:

```
helm repo add argo https://argoproj.github.io/argo-helm
```

Execute o seguinte comando para instalar/atualizar o Argo Rollouts:

```
helm upgrade --install argo-rollouts argo/argo-rollouts --namespace=deployment-system --values=caminho/values.yaml
```

Você pode especificar a localização do arquivo values.yaml (--values) ou usar o parâmetro --set para alterar o valor padrão das variáveis, por exemplo:

```
helm upgrade --install argo-rollouts argo/argo-rollouts --namespace=deployment-system --set dashboard.enabled=true
```

Instalação do plugin do kubectl

Linux

Para baixar a última versão estável execute o seguinte comando no terminal:

```
curl -LO https://github.com/argoproj/argo-rollouts/releases/latest/download/kubectl-argo-rollouts-linux-amd64
```

Observação: Para obter o binário de uma versão específica, acesse a seção de Releases conforme demonstrado na seção de introdução.

Altere a permissão do binário para permitir a sua execução:

```
chmod +x kubectl-argo-rollouts-linux-amd64
```

Mova o novo binário para a pasta /usr/local/bin/:

```
sudo mv kubectl-argo-rollouts-linux-amd64 /usr/local/bin/kubectl-argo-rollouts
```

Para testar se o plugin foi instalado com sucesso, execute o comando:

```
kubectl argo rollouts version
```

Você obterá uma saída similar a esta:

```
[c99303a@VDICTXFDRH8151 ~]$ kubectl argo rollouts version
kubectl-argo-rollouts: v1.6.6+737ca89
  BuildDate: 2024-02-13T15:39:31Z
  GitCommit: 737ca89b42e4791e96e05b438c2b8540737a2a1a
  GitTreeState: clean
  GoVersion: go1.20.14
  Compiler: gc
  Platform: linux/amd64
```

Você poderá verificar os comandos disponíveis através da linha de comando:

```
kubectl argo rollouts --help
```

```
[c99303a@VDICTXFD8151 ~]$ kubectl argo rollouts --help
This command consists of multiple subcommands which can be used to manage Argo Rollouts.

Usage:
  kubectl-argo-rollouts COMMAND [flags]
  kubectl-argo-rollouts [command]

Examples:
  # Get guestbook rollout and watch progress
  kubectl argo rollouts get rollout guestbook -w

  # Pause the guestbook rollout
  kubectl argo rollouts pause guestbook

  # Promote the guestbook rollout
  kubectl argo rollouts promote guestbook





  # Abort the guestbook rollout
  kubectl argo rollouts abort guestbook

  # Retry the guestbook rollout
  kubectl argo rollouts retry guestbook

Available Commands:
  abort          Abort a rollout
  completion     Generate completion script
  create         Create a Rollout, Experiment, AnalysisTemplate, ClusterAnalysisTemplate, or AnalysisRun resource
  dashboard     Start UI dashboard
  get           Get details about rollouts and experiments
  help          Help about any command
  lint          Lint and validate a Rollout
  list          List rollouts or experiments
  notifications  Set of CLI commands that helps manage notifications settings
  pause         Pause a rollout
  promote       Promote a rollout
  restart       Restart the pods of a rollout
  retry        Retry a rollout or experiment
  set          Update various values on resources
  status       Show the status of a rollout
  terminate    Terminate an AnalysisRun or Experiment
  undo         Undo a rollout
  version      Print version

Flags:
  --as string                Username to impersonate for the operation. User could be a regular user or a service account in a namespace.
  --as-group stringArray    Group to impersonate for the operation. this flag can be repeated to specify multiple groups.
  --as-namespace string     Namespace to impersonate with. Only applicable when --as (a regular user) is provided.
  --cluster string           The name of the kubeconfig cluster to use.
  --context string           The name of the kubeconfig context to use.
  --dry-run                 If set, only display the objects that would be created.
  --help                    Print this help message.
  --kubeconfig string       Path to kubeconfig file to use for CLI requests.
  --namespace string        Namespace to restrict the rollout to.
  --server string            The address of the Kubernetes API server (protocol://host:port).
  --show-manifest            Show the manifest of the rollout.
  --timeout string          Timeout for the operation.
  --verbose                 If set, show more verbose output.
  --version                 Print the version of the tool.
  -h, --help                Print this help message.
```

Além disso, o Argo Rollouts possui uma [página de documentação](#) que descreve cada comando disponível:

 **Argo Rollouts - Kubernetes Progressive Delivery Contr...** v: latest   Search  **GitHub**
v1.6.6 2.5k 791

Argo Rollouts - Kubernetes Progressive Delivery Controller

- Architecture
- Getting Started >
- Dashboard
- Rollout >
- Traffic Management >
- Analysis >
- Experiments
- Notifications >
- Kubectl Plugin >
- Overview
- Commands >
 - Rollouts**
 - Rollouts Abort
 - Rollouts Completion
 - Rollouts Create
 - Rollouts Create Analysisrun
 - Rollouts Dashboard
 - Rollouts Get
 - Rollouts Get Experiment
 - Rollouts Get Rollout
 - Rollouts Lint
 - Rollouts List
 - Rollouts List Experiments
 - Rollouts List Rollouts

Rollouts

Manage argo rollouts

Synopsis

This command consists of multiple subcommands which can be used to manage Argo Rollouts.

```
kubectl argo rollouts COMMAND [flags]
```

Examples

```
# Get guestbook rollout and watch progress
kubectl argo rollouts get rollout guestbook -w

# Pause the guestbook rollout
kubectl argo rollouts pause guestbook

# Promote the guestbook rollout
kubectl argo rollouts promote guestbook

# Abort the guestbook rollout
kubectl argo rollouts abort guestbook

# Retry the guestbook rollout
kubectl argo rollouts retry guestbook
```

Table of contents

- Synopsis
- Examples
- Options
- Available Commands

Windows

Primeiramente você deverá definir qual pasta será armazenado o binário.

Neste exemplo, o binário será salvo na pasta C:\opt\argo-rollouts-plugin.

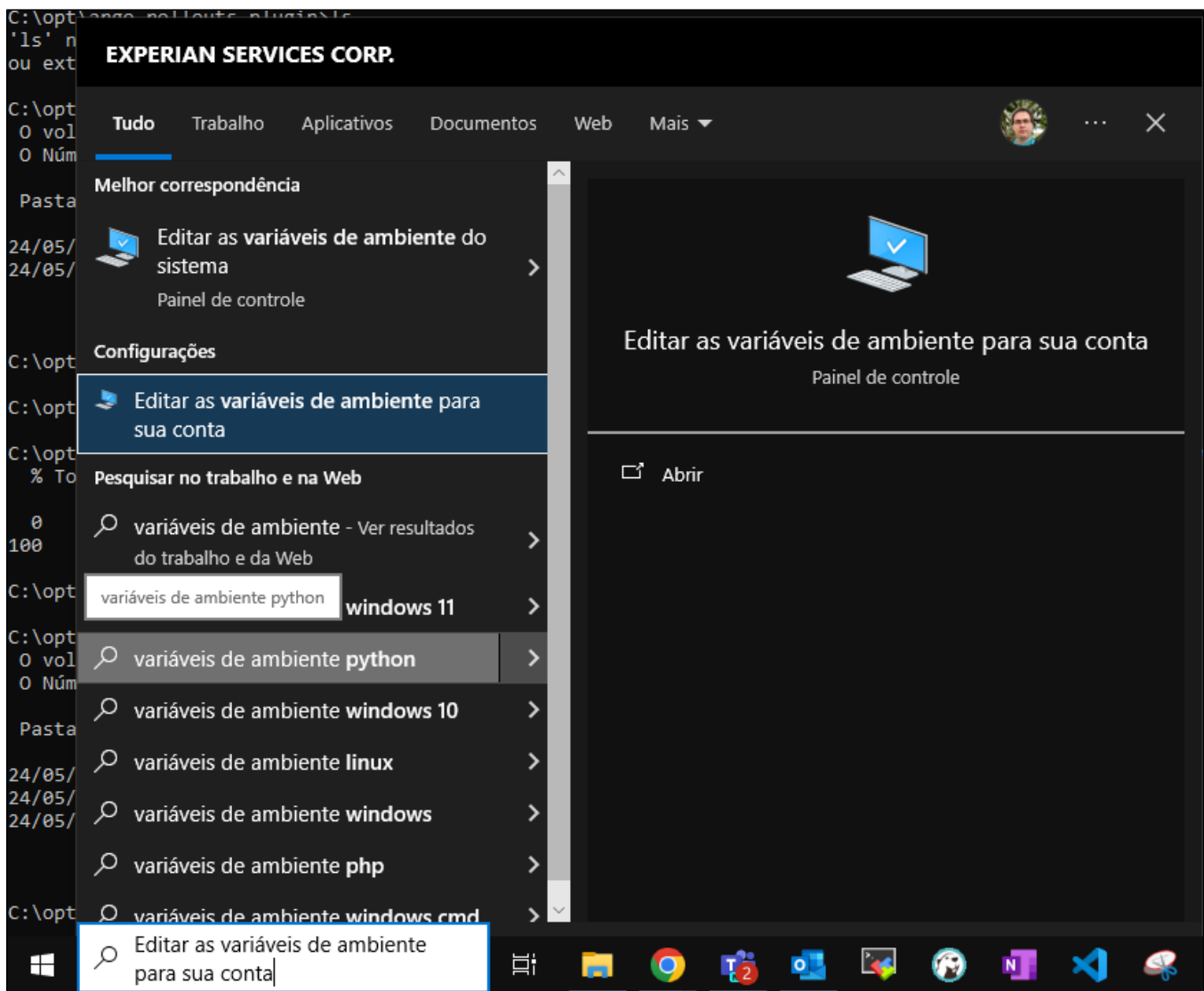
Para baixar a última versão estável execute o seguinte comando no PowerShell:

```
Invoke-WebRequest -Uri "https://github.com/argoproj/argo-rollouts/releases/latest/download/kubectl-argo-rollouts-windows-amd64" -OutFile "C:\opt\argo-rollouts-plugin\kubectl-argo-rollouts.exe"
```

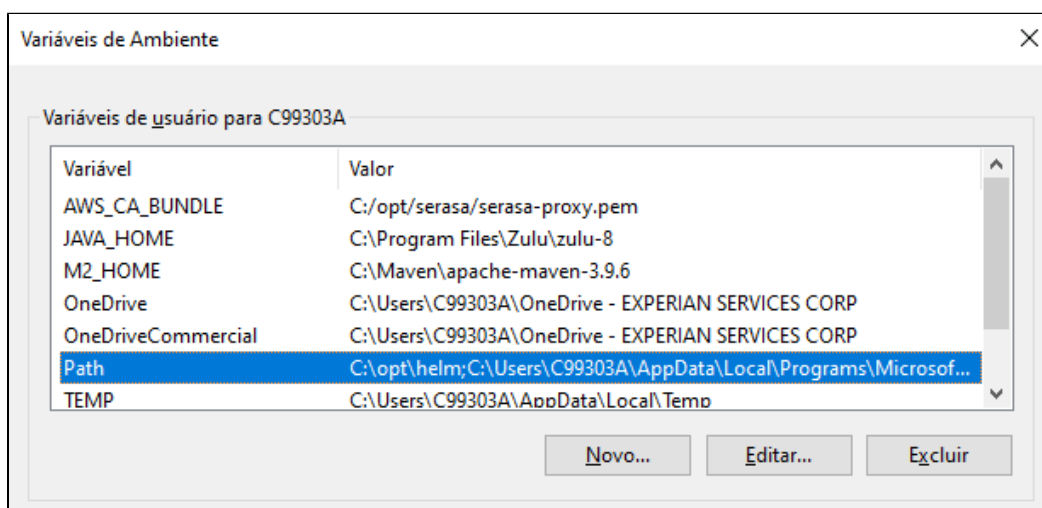
Observação: Para obter o binário de uma versão específica, acesse a seção de Releases conforme demonstrado na seção de introdução.

Pressione as teclas Win + S no teclado para abrir a barra de pesquisa do Windows.

Digite variáveis de ambiente e clique na opção "Editar as variáveis de ambiente para sua conta":

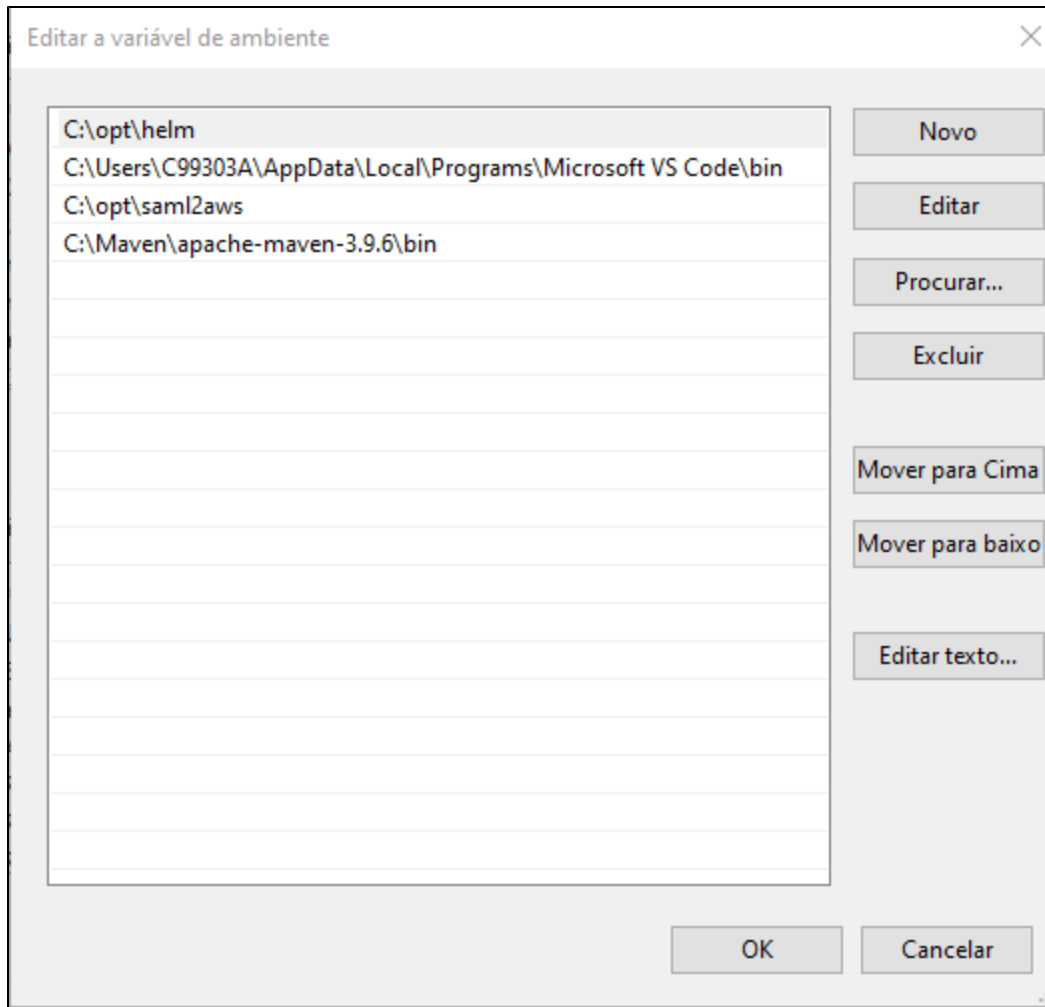


Em variáveis de usuário, procure e selecione a variável Path:



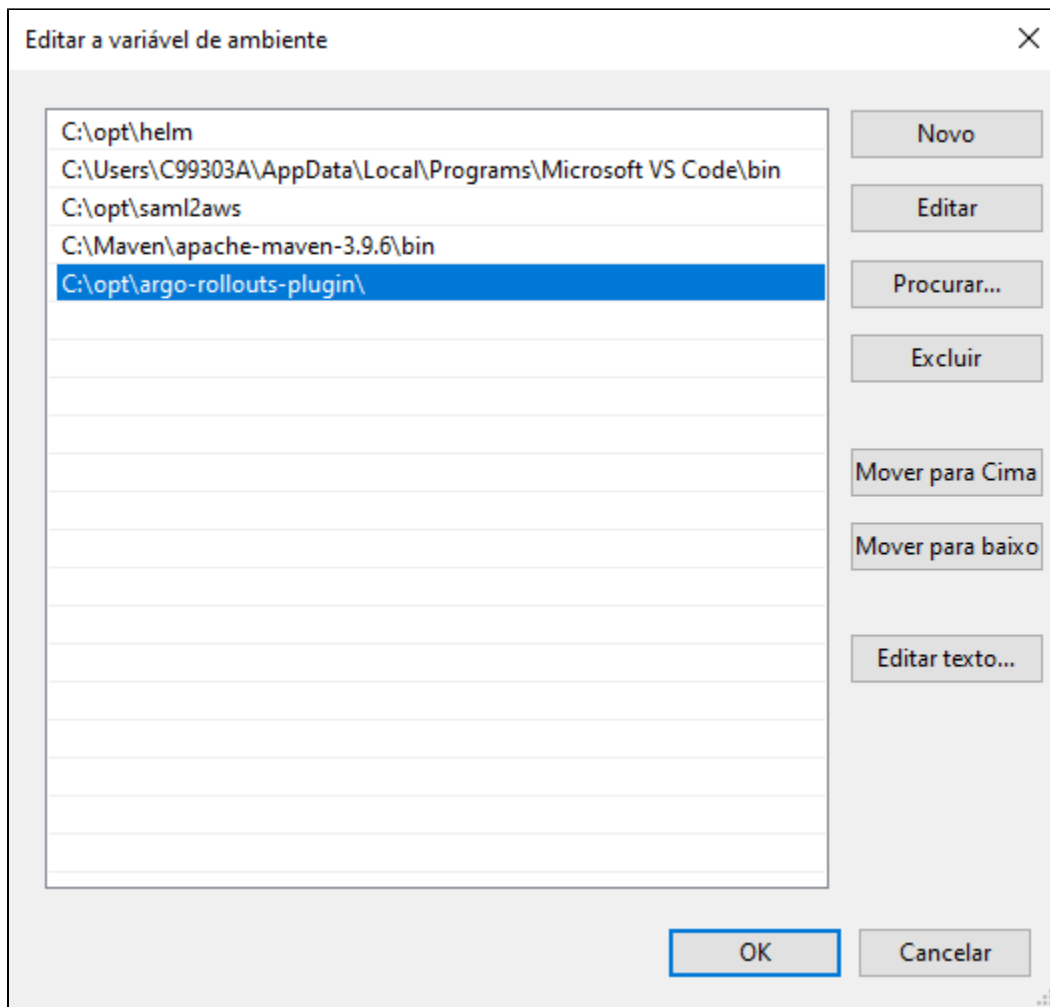
Clique no botão Editar.

Posteriormente será aberto a seguinte janela:



Clique no botão Novo.

Informe o endereço completo da pasta que contém o binário que baixamos:



Clique em OK.

Clique em OK para fechar a outra tela.

Para testar se o plugin foi instalado com sucesso, execute o seguinte comando no PowerShell ou Prompt de Comando:

```
kubectl argo rollouts version
```

Você obterá uma saída similar a esta:

```
PS C:\Users\C99303A> kubectl argo rollouts version
kubectl-argo-rollouts: v1.6.6+737ca89
  BuildDate: 2024-02-13T15:45:11Z
  GitCommit: 737ca89b42e4791e96e05b438c2b8540737a2a1a
  GitTreeState: clean
  GoVersion: go1.20.14
  Compiler: gc
  Platform: windows/amd64
```

Você poderá verificar os comandos disponíveis através da linha de comando:

```
kubectl argo rollouts --help
```

```

C:\opt\argo-rollouts-plugin>kubectl argo rollouts --help
This command consists of multiple subcommands which can be used to manage Argo Rollouts.

Usage:
  kubectl-argo-rollouts COMMAND [flags]
  kubectl-argo-rollouts [command]

Examples:
# Get guestbook rollout and watch progress
kubectl argo rollouts get rollout guestbook -w

# Pause the guestbook rollout
kubectl argo rollouts pause guestbook

# Promote the guestbook rollout
kubectl argo rollouts promote guestbook

# Abort the guestbook rollout
kubectl argo rollouts abort guestbook

# Retry the guestbook rollout
kubectl argo rollouts retry guestbook

Available Commands:
  abort          Abort a rollout
  completion     Generate completion script
  create         Create a Rollout, Experiment, AnalysisTemplate, ClusterAnalysisTemplate, or AnalysisRun resource
  dashboard      Start UI dashboard
  get            Get details about rollouts and experiments
  help           Help about any command
  lint           Lint and validate a Rollout
  list           List rollouts or experiments
  notifications   Set of CLI commands that helps manage notifications settings
  pause          Pause a rollout
  promote        Promote a rollout
  restart        Restart the pods of a rollout
  retry          Retry a rollout or experiment
  set            Update various values on resources
  status         Show the status of a rollout
  terminate      Terminate an AnalysisRun or Experiment
  undo           Undo a rollout
  version        Print version

Flags:
  --as string                Username to impersonate for the operation. User could be a regular user or a s
  --as-group stringArray     Group to impersonate for the operation, this flag can be repeated to specify m

```

Além disso, o Argo Rollouts possui uma [página de documentação](#) que descreve cada comando disponível:

Argo Rollouts - Kubernetes
Progressive Delivery
Controller

Architecture

Getting Started

Dashboard

Rollout

Traffic Management

Analysis

Experiments

Notifications

Kubectl Plugin

Overview

Commands

Rollouts

Rollouts Abort

Rollouts Completion

Rollouts Create

Rollouts Create

Analysisrun

Rollouts Dashboard

Rollouts Get

Rollouts Get Experiment

Rollouts Get Rollout

Rollouts Lint

Rollouts List

Rollouts List Experiments

Rollouts List Rollouts

Rollouts

Manage argo rollouts

Synopsis

This command consists of multiple subcommands which can be used to manage Argo Rollouts.

```
kubectl argo rollouts COMMAND [flags]
```

Examples

```
# Get guestbook rollout and watch progress
kubectl argo rollouts get rollout guestbook -w
```

```
# Pause the guestbook rollout
kubectl argo rollouts pause guestbook
```

```
# Promote the guestbook rollout
kubectl argo rollouts promote guestbook
```

```
# Abort the guestbook rollout
kubectl argo rollouts abort guestbook
```

```
# Retry the guestbook rollout
kubectl argo rollouts retry guestbook
```

Table of contents

Synopsis

Examples

Options

Available Commands