

En este tutorial, aprenderá diferentes formas de guardar sus variables de entorno y secretos de GitHub Actions que puede usar cuando sea necesario mientras trabaja con GitHub Actions.

Configuración de las variables de entorno de acciones de GitHub

Al automatizar procesos con [el flujo de trabajo de GitHub Actions](#), es posible que necesite adjuntar variables de entorno a sus flujos de trabajo. ¿Cómo? Primero debe crear y especificar variables de entorno personalizadas en el flujo de trabajo con la palabra clave ENV

1. Cree un directorio llamado `.github/workflows` donde almacenará su archivo de flujo de trabajo.

2. A continuación, cree un archivo con su nombre preferido en el directorio `.github/workflows`. Pero para este ejemplo, el archivo se llama `main.yml`. Copie y pegue el código siguiente en el archivo `main.yml`.

El siguiente código establece y muestra la variable de entorno `API_KEY`.

```
name: env_tutorial

## Triggers the workflow on when there is a push, or
## pull request on the main branch
on: [pull_request, push]

env:
  ## Sets environment variable
  API_KEY: XXXXXXXXXXXXX

jobs:
  job1:
    ## The type of runner that the job will run on,
```

```

## here it runs on ubuntu latest
runs-on: ubuntu-latest
steps:
  - name: step 1
    ## Reference your environment variables
    run: echo "The API key is:${{env.API_KEY}}"

job2:
  runs-on: ubuntu-latest
  steps:
    - name: step 1
      ## Another way reference your environment
      variables
      run: echo "The API key is:$API_KEY"

```

3. Haga ahora push al repositorio

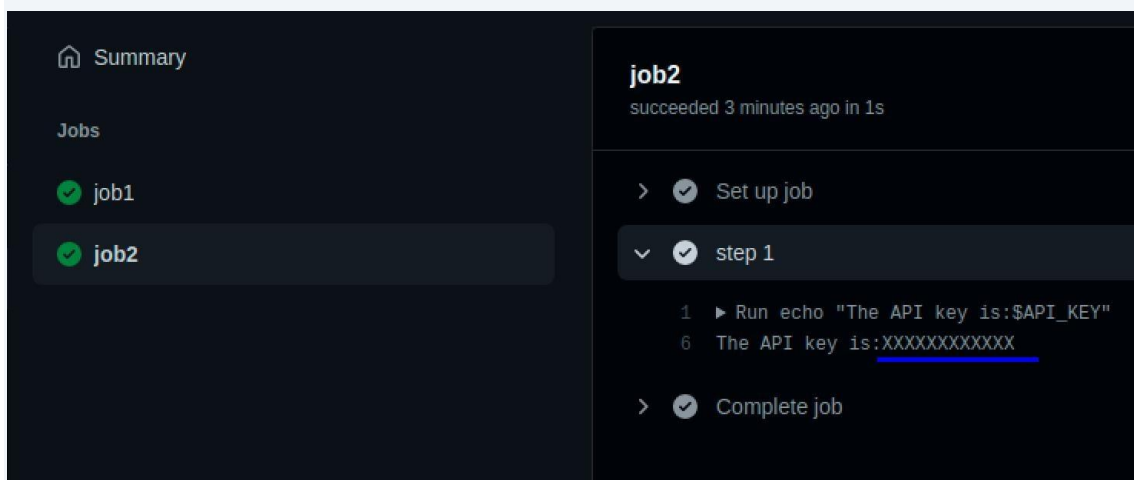
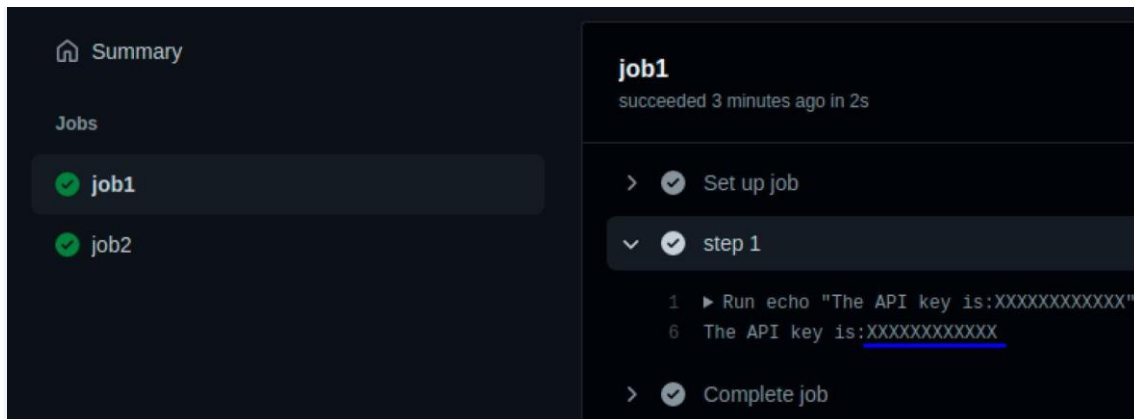
4. Ahora, abra su navegador web y navegue hasta su proyecto en GitHub. Haga clic en la pestaña **Acciones** , luego haga clic en su confirmación actual.

Verá algo como la imagen a continuación, que muestra que GitHub ha ejecutado el flujo de trabajo.

The screenshot shows the GitHub Actions interface for a workflow named 'initial commit env_tutorial #3'. The status is 'Success'. The workflow was triggered by a push to the master branch by user 'khabdrick' 1 minute ago. The summary shows two jobs, 'job1' and 'job2', both of which completed successfully in 1 second each. The workflow configuration file 'main.yml' is shown with the trigger 'on: push'.

Job	Status	Duration
job1	Success	1s
job2	Success	1s

5. Finalmente, haga clic en **trabajo1** o **trabajo2** y verá que ha hecho referencia con éxito a la variable de entorno que inicializó.



Definición de una variable de entorno para un trabajo

Ahora que ha inicializado la variable de entorno en todo el archivo de flujo de trabajo, cualquier trabajo puede hacer referencia a la variable de entorno. Pero quizás solo desee que un trabajo haga referencia a la variable de entorno. Si es así, coloque la `env` palabra clave en el trabajo mismo.

1. Reemplace el código en su archivo *main.yml* con el código a continuación.

El siguiente código muestra que cuando coloca la variable de entorno en un trabajo en particular, otros trabajos no pueden hacer referencia a la variable de entorno.

```
name: env_tutorial

## Triggers the workflow on when there is a push, or
## pull request on the main branch
on: [pull_request, push]

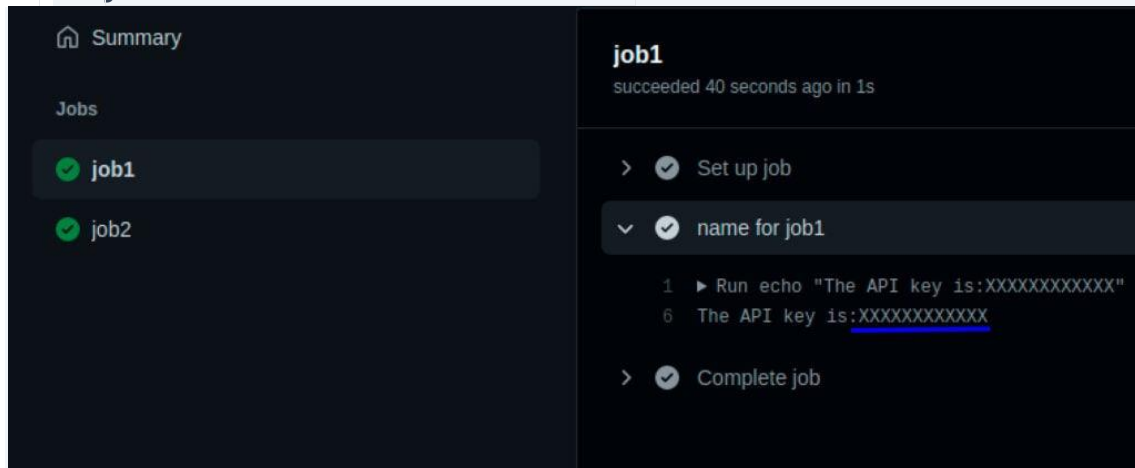
jobs:
  job1:
    ## The type of runner that the job will run on
    runs-on: ubuntu-latest
    env:
      ## Environment variable
      API_KEY: XXXXXXXXXXXX
    steps:
      - name: step 1
        ## Reference your environment variables
        run: echo "The API key is:${{env.API_KEY}}"

  job2:
    runs-on: ubuntu-latest
    steps:
      - name: step 1
        ## Another way reference your environment
        variables
        run: echo "The API key is:$API_KEY"
```

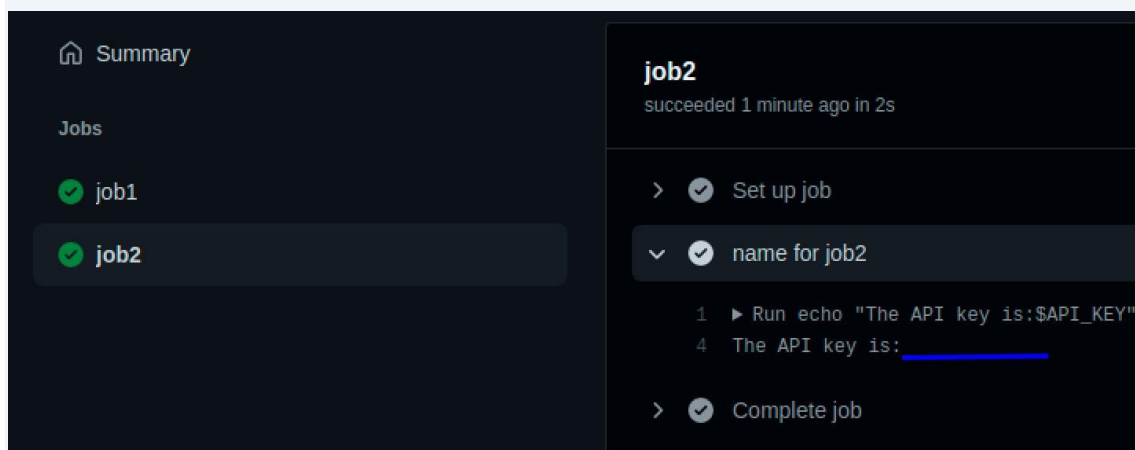
2. Confirme sus cambios y envíe trabajos desde su código a las variables de entorno de GitHub Actions como lo hizo en la sección anterior.

3. Finalmente, navegue a su proyecto en GitHub, luego haga clic en **job1** y **job2** para ver su comparación:

- **job1** : verá que ha hecho referencia a la variable de entorno perfectamente.
- **job2** : la clave API está en blanco.



The screenshot shows the GitHub Actions workflow summary for 'job1'. On the left, under 'Jobs', both 'job1' and 'job2' are listed with green checkmarks. The main panel for 'job1' shows it 'succeeded 40 seconds ago in 1s'. The steps are: 'Set up job', 'name for job1' (expanded), and 'Complete job'. The expanded step shows a command to run 'echo "The API key is:XXXXXXXXXX"' and the output 'The API key is:XXXXXXXXXX'.



The screenshot shows the GitHub Actions workflow summary for 'job2'. On the left, under 'Jobs', both 'job1' and 'job2' are listed with green checkmarks. The main panel for 'job2' shows it 'succeeded 1 minute ago in 2s'. The steps are: 'Set up job', 'name for job2' (expanded), and 'Complete job'. The expanded step shows a command to run 'echo "The API key is:\$API_KEY"' and the output 'The API key is:' followed by a blank line.

Definición de una variable de entorno para un paso

Ahora que ha aprendido a especificar variables de entorno dentro de un trabajo, debe preguntarse cómo puede hacer lo mismo con los pasos.

Para los pasos de un trabajo, especifique la variable de entorno dentro del paso como lo hizo para el trabajo.

1. Reemplace el código que tiene en su archivo *main.yml* con el código a continuación.

En el siguiente código, especifica la variable de entorno en **step 1** pero no en **step 2**, y verá el efecto en los siguientes pasos.

```
name: env_tutorial

## Triggers the workflow on when there is a push, or
## pull request on the main branch
on: [pull_request, push]

jobs:
  job1:
    ## The type of runner that the job will run on
    runs-on: ubuntu-latest
    steps:
      - name: step 1
        env:
          ## Environment variable for step 1
          API_KEY: XXXXXXXXXXXX
          ## Reference your environment variables
          run: echo "The API key is:${{env.API_KEY}}"
      - name: step 2
        ## Reference your environment variables
        run: echo "The API key is:${{env.API_KEY}}"
```

2. Ahora confirme los cambios y envíe el código a GitHub.

3. Finalmente, navegue hasta su proyecto en las variables de entorno de GitHub Actions y haga clic en **job1**.

Aunque haga referencia a las dos claves de API en el mismo trabajo (**job1**) en ambos pasos, **el paso 2** no pudo evaluar la clave de API (en

blanco), como se muestra a continuación. ¿Por qué? Porque no especificó la variable de entorno dentro `step 2` de su código.



```
job1
succeeded 11 seconds ago in 1s

> ✓ Set up job

v ✓ step 1
  1 ▶ Run echo "The API key is:XXXXXXXXXXXX"
  6 The API key is:XXXXXXXXXXXX

v ✓ step 2
  1 ▶ Run echo "The API key is:"
  4 The API key is: _____

> ✓ Complete job
```

Gestión de variables de entorno a través de variables de entorno y secretos de GitHub Actions

En lugar de codificar, es posible que desee almacenar su variable de entorno de forma segura, y los secretos de GitHub pueden hacer precisamente eso. Las variables de entorno de GitHub Actions encriptan los valores que colocas en los secretos, por lo que no son visibles ni legibles a simple vista.

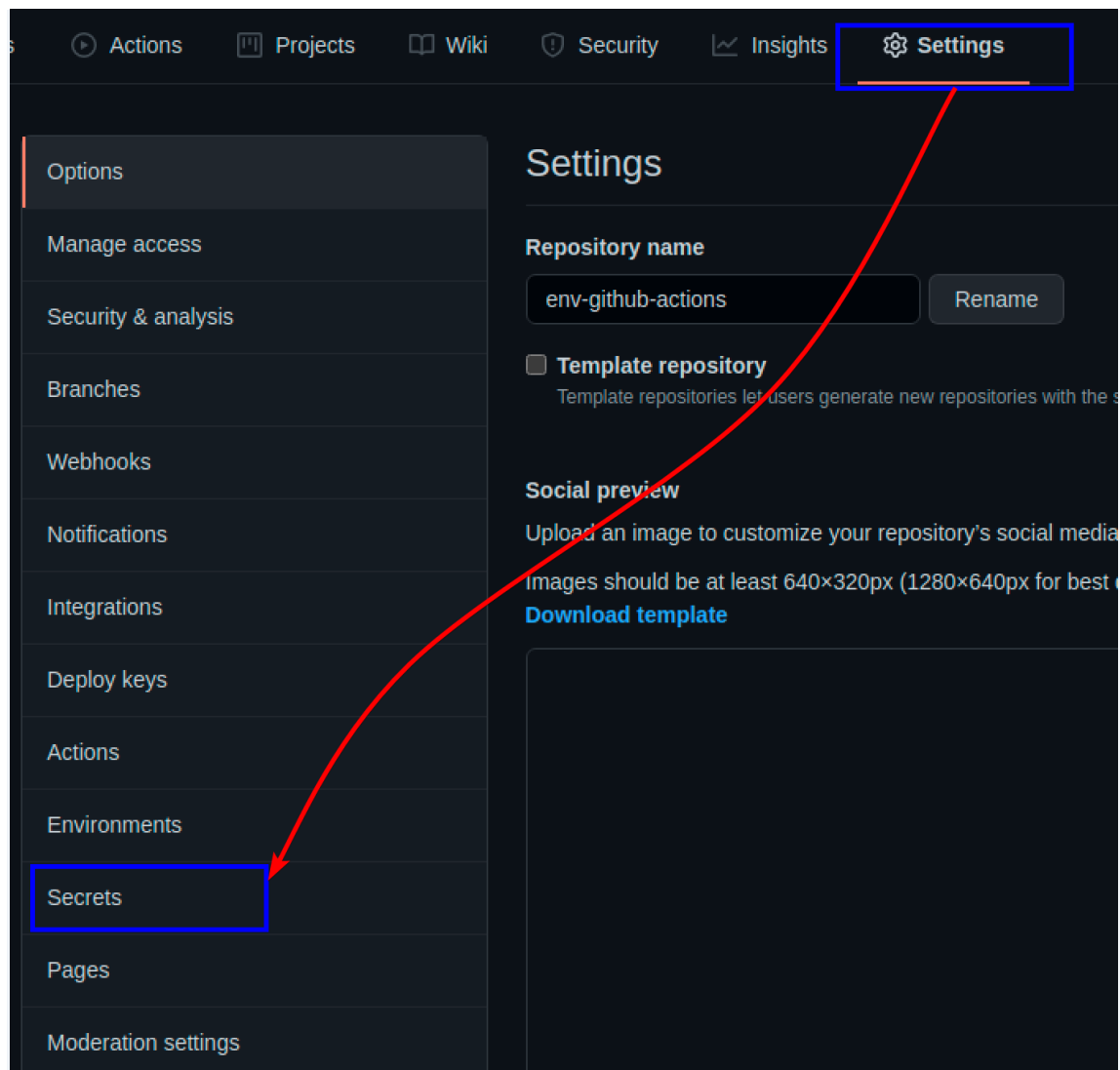
El secreto creado con este método es accesible para todo el flujo de trabajo, los trabajos y los pasos; no hay restricciones.

Para almacenar su variable de entorno en GitHub Secrets:

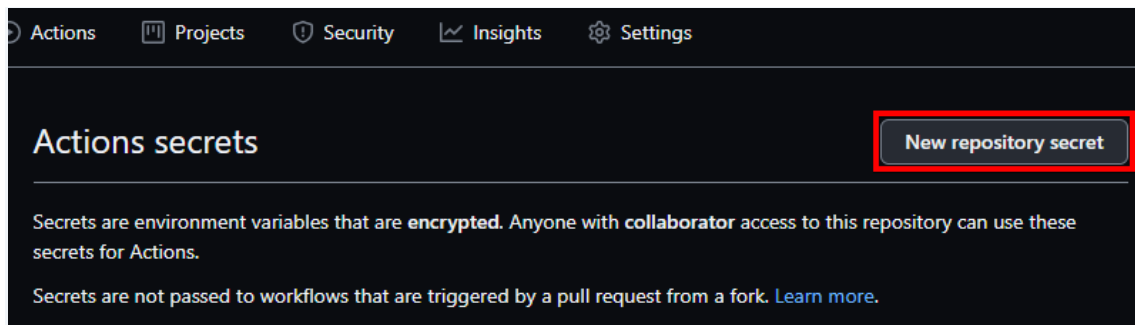
1. Primero, envíe su código a GitHub como lo hizo en las secciones anteriores.

2. A continuación, navegue hasta su proyecto en GitHub y haga clic en la pestaña **Configuración**.

Haga clic en **Secretos** en la pestaña a continuación para comenzar a agregar un secreto.



3. A continuación, haga clic en **Nuevo secreto del repositorio** y verá un formulario para completar los detalles sobre el secreto que está agregando.



4. Rellene el formulario correctamente (**Nombre** y **Valor**) y haga clic en el botón **Añadir secreto** para enviar. Ahora `API_KEY` está guardado en GitHub Secrets. De esta forma, GitHub establece de forma segura las variables de entorno como secretos a los que puede hacer referencia cuando trabaja en GitHub Actions.

Actions secrets / New secret

Name

API_KEY

Value

XXXXXXXXXXXXX|

Add secret

5. Edite su archivo `main.yml` y reemplace la `env` palabra clave con `secrets`.

A continuación, puede ver que hace referencia a la clave API en este `{{secrets.API_KEY}}` formato en lugar de codificar la clave API en sí.

```
name: env_tutorial

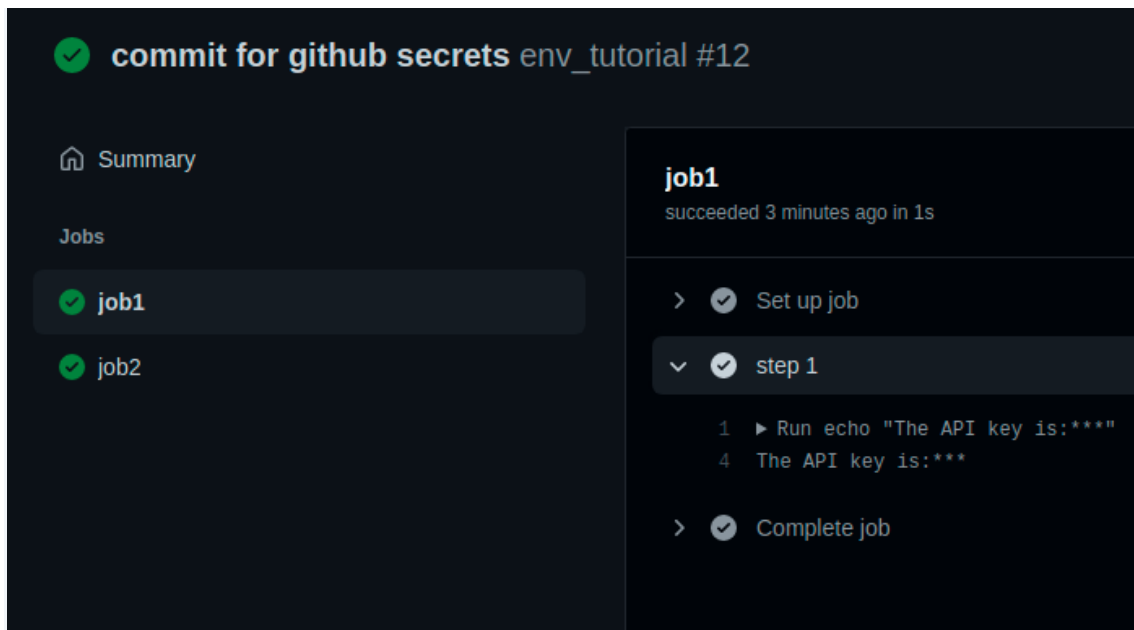
## Triggers the workflow on when there is a push, or
## pull request on the main branch
on: [pull_request, push]

jobs:
  job1:
    ## The type of runner that the job will run on
    runs-on: ubuntu-latest
    steps:
      - name: step 1
        ## Reference your environment variables
        run: echo "The API key is:{{secrets.API_KEY}}"

  job2:
    runs-on: ubuntu-latest
    steps:
      - name: step 1
        ## Reference your environment variables
        run: echo "The API key is:{{secrets.API_KEY}}"
```

6. Finalmente, confirme y envíe el código a GitHub y navegue hasta su proyecto en las variables de entorno de GitHub Actions. Referencia a la primera sección.

Verá algo como la imagen a continuación, pero no puede ver el real `API_key` ya que GitHub encripta los valores que ingresa en secretos.



Hacer referencia a las variables de entorno predeterminadas de GitHub

Hay un par de [variables de entorno predeterminadas proporcionadas por GitHub](#), que puede usar para acceder a los sistemas de archivos en el repositorio en lugar de rutas codificadas. Las variables de entorno predeterminadas de GitHub le permiten ser más dinámico al hacer referencia a las variables de entorno que le proporciona GitHub.

Algunas de las rutas que puede obtener con las variables de entorno predeterminadas son las siguientes:

- `GITHUB_JOB`– Proporciona `job_id` del trabajo actual.
- `GITHUB_ACTION`– Proporciona el id de la acción actual
- `GITHUB_ACTION_PATH`– Proporciona la ruta donde se encuentra su acción.
- `GITHUB_ACTOR`– proporciona el nombre de la persona o aplicación que inició el flujo de trabajo, como su nombre de usuario de GitHub.
- `GITHUB_RUN_ID`– proporciona el número único del `run`comando.

Reemplace lo que tiene en su archivo *main.yml* con el siguiente código. El siguiente código muestra la variable de entorno predeterminada indicada en el código.

```
name: env_tutorial

## Triggers the workflow on when there is a push or
## pull request on the main branch
on: [pull_request, push]


jobs:
  job1:
    ## The type of runner that the job will run on
    runs-on: ubuntu-latest
    steps:
      - name: step 1
        run: |
          echo "The job_id is: $GITHUB_JOB" # reference
the default environment variables
          echo "The id of this action is: $GITHUB_ACTION"
# reference the default environment variables
          echo "The run id is: $GITHUB_RUN_ID"
          echo "The GitHub Actor's username is:
$GITHUB_ACTOR"
      - name: step 2
        run: |
          echo "The run id is: $GITHUB_RUN_ID"
```

Copiar

Confirme y envíe los cambios de código a GitHub, verifique sus acciones en su proyecto de variables de entorno de GitHub Actions y verá algo como la imagen a continuación.

job1

succeeded 42 seconds ago in 1s

>  Set up job

▼  step 1

```
1 ▶ Run echo "The job_id is: $GITHUB_JOB" # referenced by workflow
7 The job_id is: job1
8 The id of this action is: __run
9 The run id is: 1477191234
10 The GitHub Actor's username is: khabdrick
```

▼  step 2

```
1 ▶ Run echo "The run id is: $GITHUB_RUN_ID"
4 The run id is: 1477191234
```

Conclusión

A lo largo de este laboratorio, ha aprendido a administrar las variables de entorno de GitHub Actions. Ahora debe tener un conocimiento básico sobre cómo almacenar variables de entorno de forma segura y cómo usar las predeterminadas proporcionadas por GitHub.