

# Uso del cliente de VMware vRealize Orchestrator

Febrero de 2022  
vRealize Orchestrator 8.7

Puede encontrar la documentación técnica más actualizada en el sitio web de VMware:

<https://docs.vmware.com/es/>

**VMware, Inc.**  
3401 Hillview Ave.  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

**VMware Spain, S.L.**  
Calle Rafael Boti 26  
2.ª planta  
Madrid 28023  
Tel.: +34 914125000  
[www.vmware.com/es](http://www.vmware.com/es)

Copyright © 2008-2022 VMware, Inc. Todos los derechos reservados. [Información sobre el copyright y la marca comercial.](#)

# Contenido

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Uso del cliente de VMware vRealize Orchestrator</b>   | <b>6</b>  |
| <b>2</b> | <b>Introducción al cliente de vRealize Orchestrator</b>  | <b>7</b>  |
|          | Panel de uso del cliente de vRealize Orchestrator  | 8         |
|          | Organización de contenido en vRealize Orchestrator Client  | 8         |
|          | Crear carpetas o subcarpetas   | 10        |
|          | Mover objetos y carpetas   | 11        |
|          | Eliminar carpetas o subcarpetas  | 11        |
| <b>3</b> | <b>Configurar vRealize Orchestrator Client</b>   | <b>13</b> |
|          | Funciones y grupos de vRealize Orchestrator  | 13        |
|          | Asignar funciones en el vRealize Orchestrator Client   | 15        |
|          | Configurar funciones de cliente de vRealize Orchestrator en vRealize Automation                                  | 16        |
|          | Crear grupos en el vRealize Orchestrator Client  | 17        |
|          | Historial de versiones de objetos de vRealize Orchestrator   | 17        |
|          | Revertir un flujo de trabajo a una versión anterior  | 18        |
|          | Comparación visual entre versiones de flujo de trabajo   | 19        |
|          | Restablecer un inventario de contenido de vRealize Orchestrator a un estado anterior con Git                     | 20        |
| <b>4</b> | <b>Casos prácticos de vRealize Orchestrator</b>  | <b>21</b> |
|          | Integrar Amazon Web Services en vRealize Orchestrator mediante Python  | 21        |
|          | Crear el script de Python inicial  | 22        |
|          | Crear la acción de Amazon Web Services   | 23        |
|          | Depurar la acción de Amazon Web Services   | 24        |
|          | Actualizar la acción de Amazon Web Services  | 28        |
|          | Cómo se puede utilizar la ramificación de Git para administrar el inventario de objetos de vRealize Orchestrator | 28        |
|          | Preparar su entorno de GitLab  | 29        |
|          | Configurar una conexión a un repositorio de Git  | 30        |
|          | Insertar cambios en un repositorio de Git  | 31        |
|          | Usar módulos de terceros para llamar a la API de proyecto de vRealize Automation                                 | 33        |
|          | Crear un script de Python que llame a la API de proyecto de vRealize Automation                                  | 33        |
|          | Crear un script de Node.js que llame a la API de proyecto de vRealize Automation                                 | 36        |
|          | Crear un script de PowerShell que llame a la API de proyecto de vRealize Automation                              | 38        |
| <b>5</b> | <b>Administrar flujos de trabajo</b>   | <b>42</b> |
|          | Flujos de trabajo estándar en la biblioteca de flujos de trabajo de vRealize Orchestrator                        | 43        |

|   |           |
|---|-----------|
| Crear flujos de trabajo   | 43        |
| Editar flujos de trabajo y acciones desde el flujo de trabajo principal   | 44        |
| Diseñador de formularios de entrada de vRealize Orchestrator  | 44        |
| Crear el cuadro de diálogo de parámetros de entrada del flujo de trabajo en el cliente de vRealize Orchestrator | 45        |
| Propiedades de parámetros de entrada en el cliente de vRealize Orchestrator                                     | 45        |
| Usar acciones para validar entradas de flujos de trabajo de vRealize Orchestrator                               | 46        |
| Solicitudes de interacción del usuario en el cliente de vRealize Orchestrator                                   | 48        |
| Programar flujos de trabajo   | 48        |
| Editar una tarea programada en el cliente de vRealize Orchestrator  | 49        |
| Buscar referencias de objetos en flujos de trabajo  | 50        |
| <b>6 Administrar acciones</b>   | <b>51</b> |
| Crear acciones  | 51        |
| Ejecutar y depurar acciones   | 52        |
| Ejecutar acciones   | 53        |
| Depurar acciones  | 53        |
| Conceptos básicos de los scripts de Python, Node.js y PowerShell  | 54        |
| Límites de tiempo de ejecución de los scripts de Python, Node.js y PowerShell                                   | 56        |
| <b>7 Administrar elementos de configuración</b>   | <b>58</b> |
| Crear elementos de configuración  | 58        |
| <b>8 Administrar políticas</b>  | <b>60</b> |
| Crear y aplicar políticas   | 60        |
| Elementos de política   | 61        |
| Administrar ejecuciones de políticas  | 62        |
| <b>9 Administrar elementos de recursos</b>  | <b>63</b> |
| <b>10 Administrar paquetes</b>  | <b>64</b> |
| Crear paquetes  | 64        |
| Exportar paquetes   | 65        |
| Importar paquetes   | 66        |
| <b>11 Solucionar problemas en el cliente de vRealize Orchestrator</b>   | <b>68</b> |
| Datos de métricas en el cliente de vRealize Orchestrator  | 68        |
| Generar perfiles de flujos de trabajo en el cliente de vRealize Orchestrator                                    | 68        |
| Usar el panel de control del sistema de vRealize Orchestrator   | 69        |
| Usar la reproducción de tokens de flujos de trabajo en el cliente de vRealize Orchestrator                      | 70        |
| Validar flujos de trabajo de vRealize Orchestrator  | 72        |

|   |    |
|---|----|
| Validar un flujo de trabajo y solucionar errores de validación en el cliente de vRealize Orchestrator | 72 |
| Depurar scripts de flujos de trabajo en el cliente de vRealize Orchestrator                           | 73 |
| Depurar flujos de trabajo por elemento de esquema   | 74 |
| Configurar un contenedor de Photon OS para paquetes de Python   | 76 |

# Uso del cliente de VMware vRealize Orchestrator

# 1

En *Uso del cliente de VMware vRealize Orchestrator* se facilita información sobre la funcionalidad y las funciones de automatización de flujos de trabajo del vRealize Orchestrator Client.

## Público objetivo

Esta información está destinada a administradores de sistemas experimentados que buscan una herramienta que les ayude a ejecutar y administrar los flujos de trabajo de vRealize Orchestrator.

# Introducción al cliente de vRealize Orchestrator

## 2

Use el vRealize Orchestrator Client para administrar los servicios y objetos de vRealize Orchestrator.

Se puede acceder a vRealize Orchestrator Client en `https://FQDN_orchestrator/orchestration-ui`.

| Elemento de la interfaz de usuario | Descripción   |
|------------------------------------|---|
| Panel de control                   | Utilice el panel de control de vRealize Orchestrator Client y la función de generación de perfiles para recopilar datos de métricas útiles sobre el entorno y los flujos de trabajo de vRealize Orchestrator.   |
| Flujos de trabajo                  | Cree, edite, programe, ejecute y elimine flujos de trabajo.   |
| Acciones                           | Cree, edite y elimine acciones. El editor de acciones admite la finalización automática de los elementos de script comunes incluidos en el explorador de API de vRealize Orchestrator.  |
| Políticas                          | Cree, edite, ejecute y elimine directivas.  |
| Paquetes                           | Cree, elimine, exporte e importe paquetes que contienen objetos de vRealize Orchestrator.   |
| Configuraciones                    | Cree, ejecute y elimine elementos de configuración.   |
| Recursos                           | Exportar, importar y actualizar elementos de recursos.  |
| Grupos                             | Los usuarios con derechos de administrador pueden asignar funciones a los usuarios del vRealize Orchestrator Client y agregarlos a grupos.  |
| Registros de auditoría             | Vea los diferentes eventos, como cuando se crea un objeto, que se registran en vRealize Orchestrator Client.  |
| Repositorios de Git                | Cree una integración con un repositorio de Git y úsela para administrar el desarrollo de flujos de trabajo y otros objetos de vRealize Orchestrator en varias implementaciones. Consulte <a href="#">Cómo se puede utilizar la ramificación de Git para administrar el inventario de objetos de vRealize Orchestrator</a> . |
| Elementos eliminados               | Restaurar los objetos de vRealize Orchestrator Client eliminados, como flujos de trabajo, acciones, políticas, elementos de configuración y elementos de recursos.  |
| Explorador de API                  | Explore los comandos de API disponibles en el vRealize Orchestrator Client.<br><b>Nota</b> La instancia de vRealize Orchestrator Client se comunica con la REST API de vRealize Orchestrator a través de un proxy REST.   |

Este capítulo incluye los siguientes temas:

- [Panel de uso del cliente de vRealize Orchestrator](#)
- [Organización de contenido en vRealize Orchestrator Client](#)

## Panel de uso del cliente de vRealize Orchestrator

El panel de control de vRealize Orchestrator Client proporciona una herramienta útil para supervisar, administrar y solucionar problemas de los flujos de trabajo de vRealize Orchestrator Client.

La información del panel de control de vRealize Orchestrator Client se distribuye en cinco paneles.

| Ventana                                    | Descripción  |
|--|--|
| Ejecuciones de flujo de trabajo            | Muestra datos visuales sobre el número de ejecuciones de flujo de trabajo que están en curso, en espera o con errores.   |
| Flujos de trabajo favoritos                | Muestra los flujos de trabajo que se han añadido a los favoritos.  |
| A la espera                                | Muestra las ejecuciones de flujo de trabajo pendientes que requieren más interacciones del usuario. Estos flujos de trabajo también se muestran en el menú de notificaciones situado en la esquina superior derecha de la interfaz de usuario. |
| Ejecuciones de flujos de trabajo recientes | Administra ejecuciones de flujos de trabajo recientes. Muestra el nombre, el estado, la fecha de inicio y la fecha de fin de la ejecución de flujo de trabajo.   |
| Requiere atención                          | Muestra las ejecuciones de flujos de trabajo con errores y las métricas de rendimiento de ejecución de los flujos de trabajo.  |

## Organización de contenido en vRealize Orchestrator Client

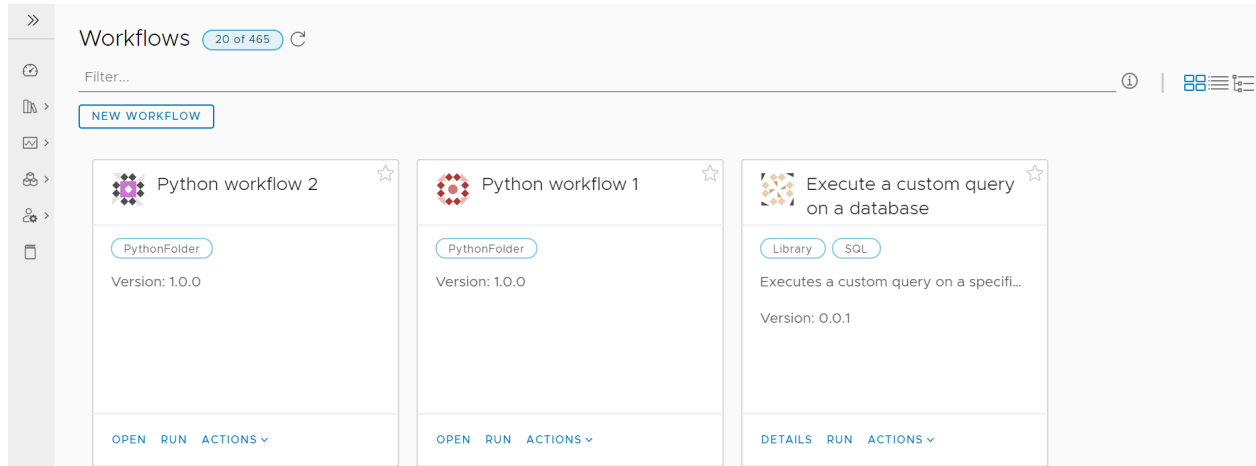
Administre cómo se muestra el inventario de objetos de vRealize Orchestrator en vRealize Orchestrator Client.

vRealize Orchestrator Client admite tres tipos distintos de vista para objetos como flujos de trabajo, acciones, directivas, recursos y configuraciones: vista de tarjeta, vista de lista y vista de árbol. El tipo de vista actual se puede cambiar en la esquina superior derecha de la página.

### Vista de tarjeta

La vista de tarjeta es el tipo de vista predeterminado que se utiliza en vRealize Orchestrator Client. La información sobre el objeto de inventario individual, como un flujo de trabajo, se muestra en un elemento de tarjeta independiente.





## Vista de lista

En la vista de lista se muestra información de los objetos de vRealize Orchestrator organizados como una lista. Para obtener más información sobre las acciones que puede realizar en el objeto, haga clic en el icono de puntos suspensivos verticales que se encuentra a la izquierda del objeto.

The screenshot shows the 'Workflows' section of the vRealize Orchestrator interface in list view. It displays a table with four columns: Name, Tags, Version, and Description. The table contains four rows of workflow information.

|   | Name                                 | Tags         | Version | Description   |
|---|--------------------------------------|--------------|---------|---|
| ⋮ | Python workflow 2                    | PythonFolder | 1.0.0   |   |
| ⋮ | Python workflow 1                    | PythonFolder | 1.0.0   |   |
| ⋮ | Execute a custom query on a database | Library SQL  | 0.0.1   | Executes a custom query on a specified database and returns the number of affected rows. You can run the workflow to update, delete, and insert queries.  |
| ⋮ | JDBC URL generator                   | Library JDBC | 0.0.8   | Solicits information to generate a connection URL for JDBC database connections. The workflow emits the connection string it generates as output via the system log, and confirms the string can create a connection to the specified database. |

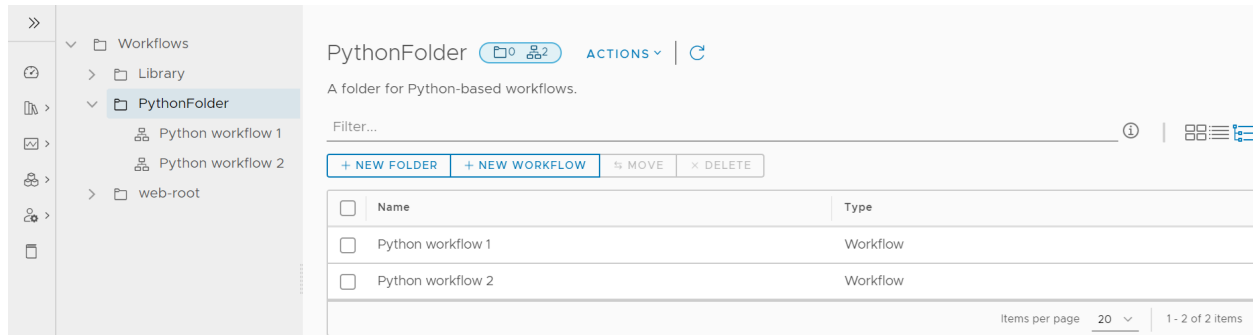
## Vista de árbol

Puede organizar el inventario de objetos en carpetas jerárquicas en la vista de árbol. Cada tipo de objeto de vRealize Orchestrator tiene una carpeta de nivel raíz. No se pueden crear objetos nuevos, como flujos de trabajo, en la carpeta raíz. Debe crear carpetas independientes organizadas en la carpeta raíz. Cada carpeta incluye herramientas que le ayudarán a administrar su contenido, como un filtro de contenido.

**Nota** Cada carpeta tiene un filtro de contenido independiente. No se puede filtrar el contenido de varias carpetas.

Para obtener más información sobre las carpetas, consulte [Crear una carpeta o una subcarpeta en el cliente de vRealize Orchestrator](#).

**Nota** Cuando se selecciona un objeto en la vista de árbol, este se abre en modo de solo lectura. Para editar el contenido del objeto, como las variables de flujo de trabajo o el esquema de flujo de trabajo, haga clic en **Editar** en el menú de opciones superior.




## Crear una carpeta o una subcarpeta en el cliente de vRealize Orchestrator

Organice sus objetos de vRealize Orchestrator mediante una estructura jerárquica de carpetas. Puede crear carpetas y subcarpetas para organizar los siguientes tipos de objetos de vRealize Orchestrator:

- Flujos de trabajo
- Acciones
- Directivas
- Elementos de configuración
- Elementos de recursos

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 En el panel de navegación de la izquierda, seleccione una página de objetos, como **Flujos de trabajo**.
- 3 En la parte superior derecha, seleccione el icono de vista de árbol ()
- 4 (opcional) Para crear una subcarpeta, seleccione una carpeta principal en la vista de árbol de la izquierda.
- 5 Haga clic en **Nueva carpeta**.
- 6 Escriba un nombre y una descripción, y haga clic en **Guardar**.
- 7 Agregue objetos o subcarpetas a la carpeta que acaba de crear.


- 8 (opcional) Para editar el nombre de la carpeta, seleccione **Acciones > Editar**.

## Mover objetos y carpetas en vRealize Orchestrator Client

Si desea reorganizar el contenido de vRealize Orchestrator, mueva el contenido a otra carpeta.

No se pueden mover acciones entre módulos de acciones ni objetos a una carpeta raíz. La carpeta raíz incluye las subcarpetas y carpetas de objetos principales, pero no se puede utilizar para almacenar objetos.

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 En el panel de navegación de la izquierda, seleccione una página de objetos, como **Flujos de trabajo**.
- 3 En la parte superior derecha, seleccione el icono de vista de árbol ()
- 4 Expanda la vista de árbol y seleccione el objeto o la carpeta que desea mover.
- 5 Arrastre el objeto o la carpeta a su nueva carpeta principal.

---

**Nota** Los objetos también se pueden mover a las carpetas nuevas directamente desde el editor de objetos. En la pestaña **Resumen**, haga clic en **Seleccionar carpeta** y seleccione la nueva carpeta principal del objeto. Otra opción para mover contenido consiste en seleccionar objetos desde la tabla en la página de la carpeta. Esta opción resulta muy útil para realizar operaciones de movimiento de lotes que incluyan varios objetos de vRealize Orchestrator.


---

## Eliminar una carpeta o una subcarpeta en vRealize Orchestrator Client

Elimine las carpetas o subcarpetas obsoletas de vRealize Orchestrator Client.

No se puede eliminar la carpeta de nivel raíz que corresponda a cada tipo de objeto de vRealize Orchestrator.

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 En el panel de navegación de la izquierda, seleccione una página de objetos, como **Flujos de trabajo**.
- 3 En la parte superior derecha, seleccione el icono de vista de árbol ()
- 4 Marque la casilla de verificación que se encuentra junto a la carpeta que desea eliminar.

---

**Nota** Para eliminar una subcarpeta, seleccione la carpeta principal en la vista de árbol y, a continuación, marque la casilla de verificación.

---

- 5 Haga clic en **Eliminar**.

- 6 Si la carpeta seleccionada está vacía.
  - a Confirme que desea eliminar la carpeta.
  - b Haga clic en **Eliminar**.
- 7 Si la carpeta seleccionada contiene objetos o subcarpetas de vRealize Orchestrator Client.
  - a Confirme que desea eliminar la carpeta.
  - b Haga clic en **Eliminar**.

Recibirá entonces el mensaje No se pudo eliminar el elemento 'nombre\_de\_carpeta': la carpeta 'nombre\_de\_carpeta' no está vacía.
  - c Para eliminar la carpeta y todo su contenido, haga clic en **Forzar eliminación**.
  - d Confirme que desea eliminar la carpeta y, a continuación, haga clic en **Eliminar**.

---

**Nota** También puede realizar una eliminación por lotes si selecciona varios objetos en la tabla que se incluye en el menú de carpetas.

---

# Configurar vRealize Orchestrator Client

## 3

Para aprovechar al máximo la funcionalidad de vRealize Orchestrator Client, debe configurar los permisos de usuario y aprender a utilizar el historial de versiones para administrar los objetos.

Este capítulo incluye los siguientes temas:

- [Funciones y grupos de vRealize Orchestrator](#)
- [Historial de versiones de objetos de vRealize Orchestrator](#)

## Funciones y grupos de vRealize Orchestrator

Los administradores de vRealize Orchestrator pueden establecer permisos que controlen el acceso a las funciones y el contenido de vRealize Orchestrator Client. Los derechos de acceso se dividen en funciones de usuario y permisos de grupo.

Las funciones controlan las características de vRealize Orchestrator Client que los usuarios pueden ver y utilizar. El acceso a la funcionalidad de administración de funciones depende del tipo de licencia del entorno de vRealize Orchestrator.

**Tabla 3-1. Acceso basado en licencias para la administración de funciones de vRealize Orchestrator**

| Licencia                   | Autenticación  |  |
|----------------------------|--|--|
|                            | vSphere  | vRealize Automation  |
| <b>vSphere</b>             | No se admite la administración de funciones.<br>Los grupos solo admiten permisos de ejecución.   |  |
| <b>vRealize Automation</b> | Administrar funciones en el cliente de vRealize Orchestrator.<br>Consulte <a href="#">Asignar funciones en el vRealize Orchestrator Client</a> . | Administrar funciones a través de <b>Administración de accesos e identidades</b> en vRealize Automation.<br>Consulte <a href="#">Configurar funciones de cliente de vRealize Orchestrator en vRealize Automation</a> . |

Los permisos de grupo controlan qué contenido de vRealize Orchestrator Client pueden ver y utilizar los usuarios, como flujos de trabajo, acciones, políticas, elementos de configuración y elementos de recurso. El acceso al contenido de vRealize Orchestrator del sistema preconfigurado (como acciones y flujos de trabajo estándar) se comparte entre todos los usuarios, a menos que se configure de otro modo mediante permisos de grupo.

Los derechos de acceso de los usuarios con funciones de administrador y visualizador no están restringidos por los permisos de grupo. Los derechos de acceso de los usuarios sin una función asignada y los usuarios con una función de diseñador de flujos de trabajo dependen del grupo que se le haya asignado. Puede ampliar los derechos de acceso de estos usuarios modificando sus permisos de grupo. De esta forma, puede organizar a los usuarios en proyectos comunes. Por ejemplo, puede crear un grupo que incluya usuarios que trabajen en el desarrollo de un complemento de vRealize Orchestrator personalizado y que les permita modificar únicamente el contenido que sea específico de su grupo.

**Tabla 3-2. Permisos de grupos y funciones de usuario de vRealize Orchestrator**

| Función                      | Derechos de acceso   |                   |
|------------------------------|--|-------------------|
| Administrador                | <p>Los administradores puede acceder a todas las funciones y al contenido del cliente de vRealize Orchestrator, incluido el contenido creado por grupos específicos. Se encarga de configurar las funciones de usuario, crear y eliminar grupos y agregar usuarios a los grupos. Los administradores no están limitados por permisos de grupo.</p> <hr/> <p><b>Nota</b> Los administradores de arrendatarios del entorno de vRealize Automation utilizado para autenticar vRealize Orchestrator tienen derechos de <b>administrador</b> de forma predeterminada.</p> |                   |
| Visualizador                 | <p>Los visualizadores tienen acceso de solo lectura a todo el contenido del cliente de vRealize Orchestrator, pero no pueden crear, editar, ejecutar o exportar contenido. Los visualizadores también pueden ver todos los grupos y el contenido de los grupos. Los visualizadores no están limitados por permisos de grupo.</p>   |                   |
| Permisos de grupo            |  |                   |
| No hay ningún grupo asignado | Ejecutar   | Ejecutar y editar |

Tabla 3-2. Permisos de grupos y funciones de usuario de vRealize Orchestrator (continuación)

| Función                          | Derechos de acceso  |  |   |
|----------------------------------|---|--|---|
| Diseñador de flujos de trabajo   | <ul style="list-style-type: none"> <li>■ Ver el contenido del sistema.</li> <li>■ Ver y ejecutar las ejecuciones propias.</li> <li>■ Crear, ejecutar, editar y eliminar su propio contenido.</li> </ul> | <ul style="list-style-type: none"> <li>■ Ver el contenido del sistema</li> <li>■ Ver y ejecutar las ejecuciones propias.</li> <li>■ Crear, ejecutar, editar y eliminar su propio contenido.</li> <li>■ Agregar contenido adicional al grupo.</li> <li>■ Ejecute el contenido del grupo, pero no puede editarlo.</li> </ul> | <ul style="list-style-type: none"> <li>■ Ver el contenido del sistema.</li> <li>■ Ver y ejecutar las ejecuciones propias.</li> <li>■ Crear, ejecutar, editar y eliminar su propio contenido.</li> <li>■ Agregar contenido adicional al grupo.</li> <li>■ Ejecutar y editar contenido del grupo.</li> </ul> <p><b>Nota</b> No está disponible para las instancias de vRealize Orchestrator autenticadas con vSphere.</p> |
| Usuario sin una función asignada | <ul style="list-style-type: none"> <li>■ Ver las ejecuciones propias.</li> </ul>  | <ul style="list-style-type: none"> <li>■ Ver y ejecutar las ejecuciones propias.</li> <li>■ Ver y ejecutar contenido de grupo.</li> </ul>  | <ul style="list-style-type: none"> <li>■ Ver y ejecutar las ejecuciones propias.</li> <li>■ Ver y ejecutar contenido de grupo.</li> </ul> <p><b>Nota</b> Para poder crear, editar y agregar contenido, los usuarios de este grupo deben tener asignada una función de diseñador de flujos de trabajo.</p> <p><b>Nota</b> No está disponible para las instancias de vRealize Orchestrator autenticadas con vSphere.</p>  |

## Asignar funciones en el vRealize Orchestrator Client

Como administrador, puede agregar usuarios al vRealize Orchestrator Client y establecer las funciones que pueden ver y utilizar.

La administración de funciones controla el acceso de los usuarios desde el proveedor de identidad de vRealize Orchestrator hasta las funciones del vRealize Orchestrator Client. La administración de funciones abarca la interfaz de usuario del vRealize Orchestrator Client y la funcionalidad de la API.

**Nota** La administración de funciones del lado del cliente solo está disponible para instancias de vRealize Orchestrator autenticadas con vSphere que utilizan una licencia de vRealize Automation. Para obtener información sobre la asignación de funciones a vRealize Orchestrator autenticadas con vRealize Automation, consulte [Configurar funciones de cliente de vRealize Orchestrator en vRealize Automation](#).

**Procedimiento**

- 1 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 2 Desplácese hasta **Administración > Administración de funciones**.
- 3 Haga clic en **Agregar**.
- 4 Busque el usuario o el grupo que desee agregar al vRealize Orchestrator Client.
- 5 Seleccione la función del usuario. Para obtener más información sobre las funciones, consulte [Funciones y grupos de vRealize Orchestrator](#).
- 6 Haga clic en **Guardar**.

## Configurar funciones de cliente de vRealize Orchestrator en vRealize Automation

Puede asignar funciones de servicio para el vRealize Orchestrator Client en la página **Administración de acceso e identidades** en vRealize Automation. Las funciones de servicio se pueden asignar para las instancias integradas del vRealize Orchestrator Client y las instancias independientes de vRealize Orchestrator autenticadas con vRealize Automation.

Las funciones de servicio de vRealize Orchestrator determinan las características de la instancia integrada del vRealize Orchestrator Client a las que pueden acceder los usuarios. Para obtener más información acerca de las funciones de vRealize Orchestrator, consulte [Funciones y grupos de vRealize Orchestrator](#).

---

**Nota** Las instancias independientes de vRealize Orchestrator autenticadas con vSphere que usan una licencia de vRealize Automation pueden asignar funciones directamente en el vRealize Orchestrator Client. Consulte [Asignar funciones en el vRealize Orchestrator Client](#).

---

**Requisitos previos**

- Compruebe que los usuarios y los grupos adecuados se importan desde una instancia de vIDM válida.
- Antes de asignar una función de servicio de vRealize Orchestrator al usuario, compruebe que este tenga asignada una función de organización en vRealize Automation. Consulte *Administrar usuarios y grupos en vRealize Automation* en *Administrar vRealize Automation*.

**Procedimiento**

- 1 En el menú desplegable del encabezado de la parte superior derecha, seleccione la opción **Administración de acceso e identidades**.
- 2 En la pestaña **Usuarios activos**, busque la dirección de correo electrónico del usuario que desea asignar a vRealize Orchestrator.
- 3 Active la casilla de verificación junto al usuario y haga clic en **Editar funciones**.
- 4 Haga clic en **Agregar acceso a servicio**.
- 5 En el menú desplegable de la izquierda, seleccione **Orchestrator**.



- 6 En el menú desplegable de la derecha, seleccione la función que desea asignar al usuario.
- 7 Haga clic en **Guardar**.

## Crear grupos en el vRealize Orchestrator Client

Como administrador, puede utilizar grupos para establecer el contenido de vRealize Orchestrator que los usuarios pueden ver y al que pueden acceder en el vRealize Orchestrator Client.

Puede usar el vRealize Orchestrator Client para establecer permisos de grupo en flujos de trabajo, acciones, directivas, elementos de configuración, elementos de recursos y paquetes de vRealize Orchestrator.

---

**Nota** Los usuarios de instancias de vRealize Orchestrator autenticados con vSphere solo pueden tener permisos de grupo **Ejecutar**.

---

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 2 Vaya a **Administración > Grupos**.
- 3 Haga clic en **Nuevo grupo**.
- 4 En la pestaña **Resumen**, añada un nombre y una descripción para el grupo.
- 5 En la pestaña **Usuarios**, haga clic en **Agregar**.
  - a Busque el usuario que desee agregar al grupo.
  - b Asigne permisos de grupo al usuario.
  - c Haga clic en **Agregar**.
- 6 En la pestaña **Elementos**, agregue objetos de vRealize Orchestrator al grupo.

---

**Nota** También puede agregar un objeto a grupos existentes al crear el objeto en el vRealize Orchestrator Client. Para agregar el objeto, seleccione el grupo en el menú desplegable **Accesible para** de la pestaña **Resumen/general** del editor de objetos.

---

- 7 Haga clic en **Guardar**.

## Historial de versiones de objetos de vRealize Orchestrator

vRealize Orchestrator Client mantiene un registro de historial de versiones de cada objeto de vRealize Orchestrator. Con el historial de versiones, puede comparar diferentes versiones de objetos de vRealize Orchestrator y revertir a una versión anterior.

vRealize Orchestrator crea un registro de historial de versiones para cada objeto de vRealize Orchestrator al guardar el objeto. Los cambios subsiguientes en el objeto de vRealize Orchestrator crean un nuevo registro de historial de versiones. Los registros de historial de versiones anteriores se conservan y se pueden utilizar para rastrear los cambios realizados en el objeto y revertirlo a una versión anterior. Al revertir un objeto a una versión anterior, se crea un nuevo registro de historial de versiones.

vRealize Orchestrator Client realiza un seguimiento del historial de versiones de los siguientes objetos de vRealize Orchestrator:

- Flujos de trabajo
- Acciones
- Paquetes
- Directivas
- Elementos de recursos
- Elementos de configuración

---

**Nota** Los flujos de trabajo generados no aparecen en el historial de versiones del flujo de trabajo. Por ejemplo, los flujos de trabajo generados por el flujo de trabajo **Generar flujos de trabajo de CRUD para una tabla** no aparecen en la pestaña **Historial de versiones** y no se pueden insertar en ningún repositorio de Git configurado. Para incluir estos flujos de trabajo en el historial de versiones de vRealize Orchestrator, duplique los flujos de trabajo generados.

---

Puede acceder al historial de versiones de un objeto desde la pestaña **Historial de versiones** de la página del editor de objetos. Si intenta editar un objeto al mismo tiempo que otro usuario, se puede producir un conflicto de combinación. Para resolver el conflicto de combinación, haga clic en **Resolver** a la derecha del mensaje de error. En la ventana **Resolver conflictos** tiene tres opciones:

- **Usar suyos.** El conflicto de combinación se resuelve utilizando los cambios realizados por el otro usuario.
- **Usar nuestros.** El conflicto de combinación se resuelve utilizando sus cambios.
- **Resolver.** El conflicto de combinación se resuelve editando el modelo de cambio que se muestra. Si el modelo proporcionado no es válido, esta opción no estará disponible.

## Revertir un flujo de trabajo a una versión anterior

Puede restaurar un flujo de trabajo a una versión guardada previamente.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Biblioteca > Flujos de trabajo** y seleccione un flujo de trabajo.
- 3 Seleccione la pestaña **Historial de versiones**.

- 4 Para ver una comparación entre las versiones, seleccione una versión del flujo de trabajo y seleccione otra versión en el menú desplegable **Comparar con**.

Se abre una ventana que muestra las diferencias entre la versión de flujo de trabajo actual y la versión de flujo de trabajo seleccionada.

- 5 Para restaurar el flujo de trabajo a otra versión, haga clic en **Restaurar**.

El estado del flujo de trabajo se revierte al estado de la versión seleccionada.

---

**Nota** También puede restaurar una versión del flujo de trabajo desde la vista de la herramienta de diferencias gráficas. Consulte [Comparación visual entre versiones de flujo de trabajo](#).

---

## Comparación visual entre versiones de flujo de trabajo

Compare los cambios que existen entre las versiones de flujo de trabajo con la herramienta de diferencias gráficas.

De forma predeterminada, el historial de versiones de vRealize Orchestrator muestra las diferencias entre las versiones de flujo de trabajo en un formulario YAML. También puede realizar una comparación visual entre las diferentes versiones de flujo de trabajo. Puede ver los cambios que existen en los siguientes elementos:

- La información general del flujo de trabajo, como el número de versión y la descripción del flujo de trabajo.
- Las variables que se utilizan en el flujo de trabajo.
- Los parámetros de entrada y salida del flujo de trabajo.
- El esquema del flujo de trabajo.

### Requisitos previos

Cree un flujo de trabajo.

### Procedimiento

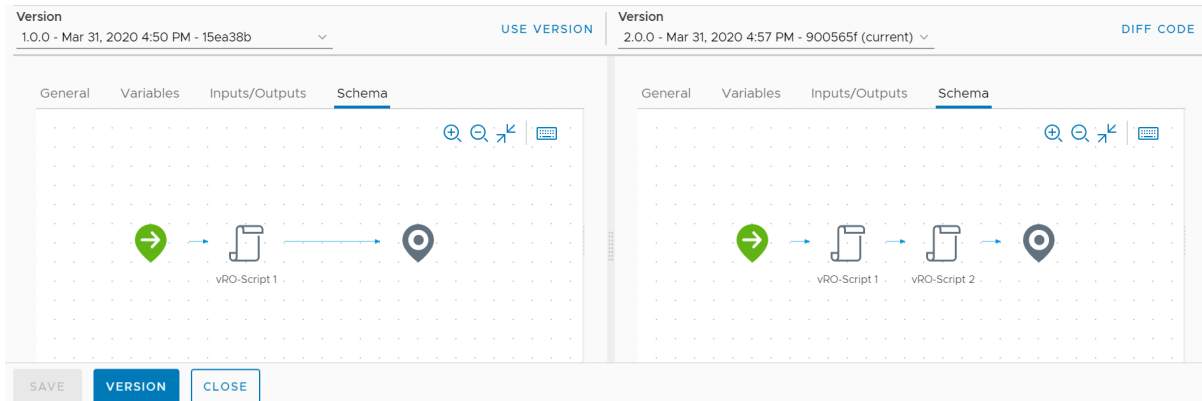
- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 Desplácese hasta **Biblioteca > Flujos de trabajo** y seleccione uno de los flujos de trabajo.
- 3 Edite el contenido del flujo de trabajo.

Por ejemplo, puede agregar un elemento **Tarea de scripts** adicional en la pestaña **Esquema**.

- 4 Haga clic en **Guardar**.
- 5 Seleccione la pestaña **Historial de versiones**.

- 6 En la parte superior derecha, seleccione **Comparar visualmente**.

Ahora podrá realizar una comparación visual entre dos versiones de flujo de trabajo seleccionadas. Puede seleccionar las versiones que desea comparar en el menú desplegable **Versión**.



- 7 (opcional) Si desea restaurar un flujo de trabajo a otra versión, seleccione **Usar versión**.

## Restablecer un inventario de contenido de vRealize Orchestrator a un estado anterior con Git

Si se usa una confirmación de Git anterior, el contenido de vRealize Orchestrator se puede restablecer a un estado anterior.

Para restablecer el contenido de vRealize Orchestrator a un estado anterior, hay que seleccionar una confirmación específica.

### Requisitos previos

- Configure una conexión a un repositorio de GitLab o GitHub. Consulte [Configurar una conexión a un repositorio de Git](#).
- Inserte un conjunto de cambios locales en el repositorio de Git configurado.

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 Vaya a **Administración > Historial de Git**.
- 3 Seleccione un conjunto de cambios al que desee restablecer y haga clic en **Restablecer a esto**.
- 4 Confirme que desea restablecer a esta confirmación específica y haga clic en **Aceptar**.

El inventario de contenido de vRealize Orchestrator se restablece al estado especificado en la confirmación. El contenido de vRealize Orchestrator relevante se revierte a una versión anterior. Si el contenido no existía cuando se insertó la confirmación, se elimina del inventario.

### Pasos siguientes

Para restaurar el inventario de vRealize Orchestrator al estado que se guardó por última vez en el repositorio de Git, ejecute un comando `Pull` en la ventana **Historial de Git**.

# Casos prácticos de vRealize Orchestrator

# 4

Estos casos prácticos demuestran parte de la funcionalidad de la plataforma de vRealize Orchestrator.

Estos casos prácticos solo presentan valores de ejemplo. La estructura de su propio entorno y las convenciones de nomenclatura pueden variar.

Este capítulo incluye los siguientes temas:

- [Integrar Amazon Web Services en vRealize Orchestrator mediante Python](#)
- [Cómo se puede utilizar la ramificación de Git para administrar el inventario de objetos de vRealize Orchestrator](#)
- [Usar módulos de terceros para llamar a la API de proyecto de vRealize Automation](#)

## Integrar Amazon Web Services en vRealize Orchestrator mediante Python

En este caso práctico de vRealize Orchestrator se muestra un ejemplo de cómo se puede utilizar Python para expandir las capacidades de la implementación de vRealize Orchestrator.

Puede utilizar los siguientes tiempos de ejecución en sus scripts de acción y flujo de trabajo:

- Python 3.7
- Node.js 14
- PowerCLI 11/Powershell 6.2
- PowerCLI 12.3.0/PowerShell 7.1

---

**Nota** El tiempo de ejecución de PowerCLI incluye PowerShell y los siguientes módulos: VMware.PowerCLI, PowerNSX y PowervRA.

---

---

**Importante** Solo puede utilizar los nuevos tiempos de ejecución si la implementación de vRealize Orchestrator utiliza una licencia de vRealize Automation.

---

En este caso práctico se muestra cómo crear un script de Python que llame a las instancias de EC2 en Amazon Web Services (AWS).

---

**Importante** Antes de comenzar a desarrollar el script personalizado, asegúrese de que está familiarizado con los conceptos básicos del uso de scripts de Python, Node.js y PowerShell en vRealize Orchestrator. Consulte [Conceptos básicos de los scripts de Python, Node.js y PowerShell](#).

---

## Procedimiento

### 1 Crear el script de Python inicial

En su ordenador local, cree el script de Python y cree un paquete con el script y una biblioteca boto3 como una carpeta ZIP.

### 2 Crear la acción de Amazon Web Services

Cree una acción de vRealize Orchestrator que utilice su script de Python.

### 3 Depurar la acción de Amazon Web Services

La versión original del script de Python tiene un error integrado deliberado, por lo que puede aprender a depurar el script.

### 4 Actualizar la acción de Amazon Web Services

Importe el script de Python actualizado y vuelva a ejecutar la acción.

## Crear el script de Python inicial

En su ordenador local, cree el script de Python y cree un paquete con el script y una biblioteca boto3 como una carpeta ZIP.

### Requisitos previos

- Descargue e instale Python 3. Consulte la [página de descargas de Python](#).
- Descargue e instale Visual Studio Code. Consulte la [página de descarga de Visual Studio Code](#).
- Compruebe que haya instalado la extensión de Python para Visual Studio Code. Consulte el [catálogo de Visual Studio](#).

### Procedimiento

- 1 En su ordenador local, cree una carpeta `vro-python-aws` e instale en ella el SDK de Python boto3.

```
mkdir vro-python-aws
cd vro-python-aws
mkdir lib
pip install boto3 -t lib/
```

- 2 Abra un editor y cree el script principal de Python. En este caso práctico, está utilizando Visual Studio Code.

```
import boto3

def handler(context, inputs):
    ec2 = boto3.resource('ec2')
    filters = [{
        'Name': 'instance-state-name',
        'Values': ['running']
    }]

    instances = ec2.instances.filter(Filters=filters)
    for instance in instances:
        print('Instance: ' + instance.id)
```

Este script de Python enumera todas las instancias de EC2 que están en ejecución en una región determinada.

- 3 Guarde el script creado como un archivo `main.py` en la carpeta `vro-python-aws`.
- 4 Inicie sesión en la interfaz de línea de comandos.
- 5 Acceda a la carpeta `vro-python-aws`.

```
cd vro-python-aws
```

- 6 Cree un paquete ZIP que contenga el script de Python.

```
zip -r --exclude=*.zip -X vro-python-aws.zip .
```

---

**Nota** También puede crear el paquete ZIP mediante una herramienta de utilidad ZIP, como 7-Zip.

---

## Resultados

Ha creado el script base de Python y lo ha preparado para importarlo en la implementación de vRealize Orchestrator.

## Crear la acción de Amazon Web Services

Cree una acción de vRealize Orchestrator que utilice su script de Python.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Vaya a **Biblioteca > Acciones**.
- 3 Haga clic en **Nueva acción**.
- 4 En la pestaña **General**, añada un nombre, un módulo y un número de versión para la acción.

- 5 En la pestaña **Script**, seleccione **Python 3.7** como el tiempo de ejecución y **Zip** como el tipo de script.
- 6 Haga clic en **Importar**.
- 7 Desplácese hasta la carpeta `vro-python-aws` y seleccione el paquete ZIP que contiene el script de Python.
- 8 En el cuadro de texto **Controlador de entrada**, introduzca `main.handler`.

---

**Nota** El controlador de entrada de la acción se basa en el script principal del paquete ZIP importado. Como el script principal se encuentra en un archivo llamado `main.py` y una función denominada `handler`, el controlador de entrada debe ser `main.handler`. Si ha nombrado a su archivo de script principal de alguna otra forma, cambie el valor del controlador de entrada como corresponda.

---

- 9 Guarde la acción y haga clic en **Ejecutar**.

Se produce un error en la ejecución de la acción.

- 10 Seleccione la pestaña **Registros**.

Los registros de la ejecución de la acción muestran un mensaje de error

`"botocore.exceptions.NoRegionError: Debe especificar una región"`. Este es el comportamiento esperado, ya que el script inicial de Python no define una región.

#### Pasos siguientes

Depure el script de Python. Consulte [Depurar la acción de Amazon Web Services](#).

## Depurar la acción de Amazon Web Services

La versión original del script de Python tiene un error integrado deliberado, por lo que puede aprender a depurar el script.

#### Requisitos previos

Inicie sesión en la cuenta de Amazon Web Services (AWS) y cree un usuario de IAM específicamente para este escenario de caso práctico. Consulte [Crear un usuario de IAM en su cuenta de AWS](#). El usuario de IAM debe tener los siguientes permisos:

```
"Effect": "Allow",
"Action": "ec2:DescribeInstances",
"Resource": "*"

```



## Procedimiento

### 1 Prepare vRealize Orchestrator Appliance.

---

**Precaución** No depure los scripts de su implementación de producción de vRealize Orchestrator. Depure desde una implementación de vRealize Orchestrator de un solo nodo, que utiliza para el desarrollo y las pruebas.

---

- a Inicie sesión en la línea de comandos de vRealize Orchestrator Appliance a través de SSH como **raíz**.
- b Ejecute el comando `vracli dev tools`.
- c Se le pedirá que confirme que desea continuar. Introduzca **yes** para continuar o **no** para cancelar.

---

**Importante** Al ejecutar el comando `vracli dev tools`, se abrirán los puertos necesarios para depurar el script de Python. Debe dejar abierta la sesión actual de SSH durante el proceso de depuración.

---

### 2 Inicie la configuración de depuración.

- a Inicie sesión en el cliente de vRealize Orchestrator.
- b Abra la acción de AWS y haga clic en **Depurar**.  
Se iniciará el proceso de depuración y se suspenderá la ejecución de la acción.
- c Seleccione la pestaña **Configuración de depuración**.  
La pestaña contiene una configuración `.json` que puede asociar de forma remota al IDE para depurar el script de Python.
- d Copie el contenido de configuración de forma manual o haga clic en **Copiar en el portapapeles**.

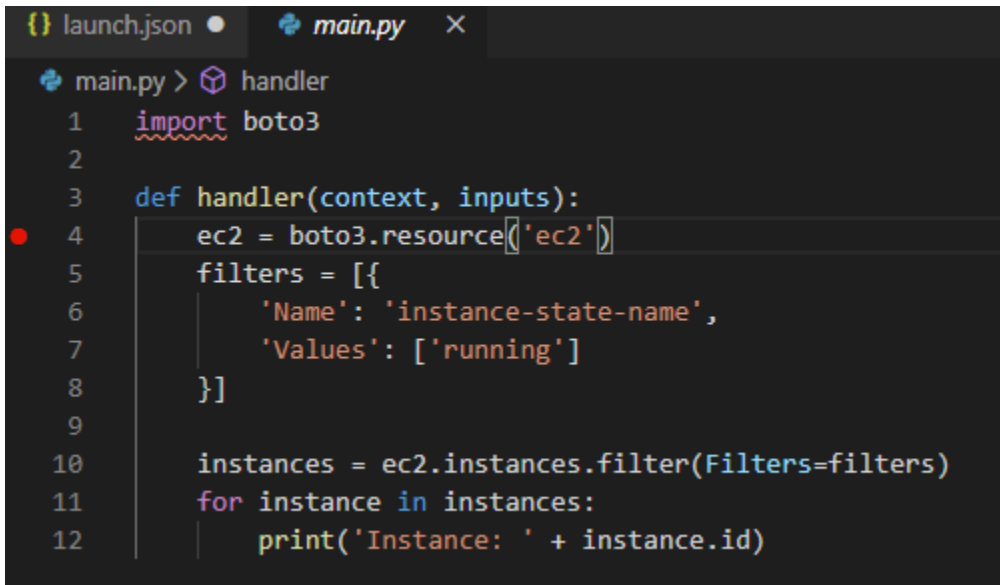
### 3 Depure el script de Python.

- a Abra Visual Studio Code.
- b Abra la carpeta `vro-python-aws`.
- c En el panel de navegación superior, seleccione **Ejecutar > Abrir configuraciones**.
- d Seleccione **Archivo de Python**.

- e Deje los atributos "version" y "configuration" en sus posiciones actuales y pegue el contenido de la configuración .json que copió del cliente de vRealize Orchestrator. El archivo launch.json que se genera debe tener un aspecto similar al siguiente:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "request": "attach",
      "port": 18281,
      "name": "vRO Python debug 8302f4c7-5beb-40da-848a-5003c0296f7b",
      "host": "es-sof-vc-vm-225-190.sof-mbu.eng.vmware.com",
      "type": "python",
      "pathMappings": [
        {
          "localRoot": "${workspaceFolder}",
          "remoteRoot": "/var/run/vco-polyglot/function"
        }
      ]
    }
  ]
}
```

- f Seleccione el archivo de script main.py y agregue un punto de interrupción a la línea de `ec2 = boto3.resource('ec2')`.



```
{ } launch.json  main.py X
main.py > handler
1  import boto3
2
3  def handler(context, inputs):
4  ec2 = boto3.resource('ec2')
5  filters = [{
6      'Name': 'instance-state-name',
7      'Values': ['running']
8  }]
9
10 instances = ec2.instances.filter(Filters=filters)
11 for instance in instances:
12     print('Instance: ' + instance.id)
```

- g En el panel de navegación superior, seleccione **Ejecutar > Iniciar depuración**.

- h Cuando el depurador llegue al punto de interrupción, realice una operación de paso a paso por procedimientos.

La ejecución de la depuración indica que el script de Python no tiene una región especificada ni una clave de acceso de AWS.

- i Regrese a la sesión de vRealize Orchestrator Appliance abierta y presione **Intro** para cerrar los puertos que abrió para esta sesión de depuración.

#### 4 Agregue la información que falta al script de Python.

- a En Visual Studio Code, cree un archivo llamado `awsconfig` que contenga la clave de acceso de AWS del usuario de IAM y la región de AWS en la que desee hacer ping con el script de Python.

```
[default]
aws_access_key_id=your key ID
aws_secret_access_key=your secret access key
region=your-region
```

- b Guarde `awsconfig` como archivo de configuración (`.cfg`) en la carpeta `vro-python-aws`.
- c Abra el archivo `main.py` y modifíquelo de manera que la biblioteca `boto3` pueda utilizar el archivo `awsconfig.cfg`.

```
import boto3

import os
os.environ['AWS_CONFIG_FILE'] = os.getcwd() + '/awsconfig.cfg'

def handler(context, inputs):
    ec2 = boto3.resource('ec2')
    filters = [{
        'Name': 'instance-state-name',
        'Values': ['running']
    }]

    instances = ec2.instances.filter(Filters=filters)
    for instance in instances:
        print('Instance: ' + instance.id)
```

- d Cree un nuevo paquete ZIP que contenga el archivo `main.py`, el archivo `awsconfig.cfg` y la biblioteca `boto3`.

```
zip -r --exclude=*.zip -X vro-python-aws.zip .
```

---

**Nota** También puede crear el paquete ZIP mediante una herramienta de utilidad ZIP, como 7-Zip.

---

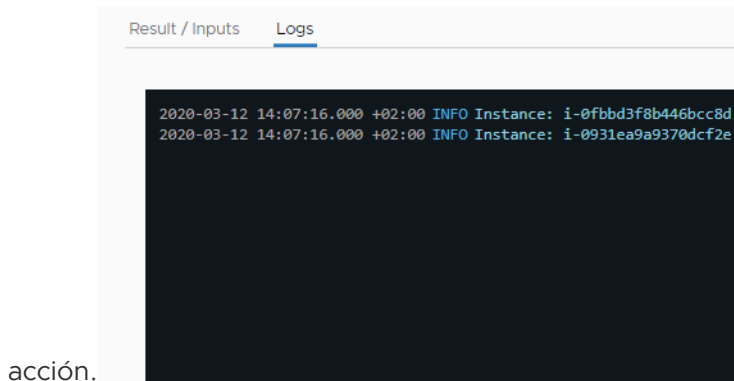
## Actualizar la acción de Amazon Web Services

Importe el script de Python actualizado y vuelva a ejecutar la acción.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Biblioteca > Acciones** y seleccione la acción original de Amazon Web Services (AWS).
- 3 (opcional) En la pestaña **General**, cambie el número de versión.
- 4 Elimine el paquete ZIP anterior y haga clic en **Importar**.
- 5 Seleccione el paquete ZIP actualizado.
- 6 Guarde la acción y haga clic en **Ejecutar**.
- 7 Una vez finalizada la ejecución de la acción, seleccione la pestaña **Registros**.

Los registros muestran las instancias de EC2 consultadas por la



### Pasos siguientes

Cree un flujo de trabajo de vRealize Orchestrator que utilice la acción de AWS actualizada como un **elemento de acción**.

## Cómo se puede utilizar la ramificación de Git para administrar el inventario de objetos de vRealize Orchestrator

Utilice la ramificación para organizar la manera en la que vRealize Orchestrator administrará el contenido en el repositorio de Git.

Con Git, aumentará la flexibilidad de sus desarrolladores de vRealize Orchestrator al proporcionar un repositorio centralizado. Por ejemplo, puede usar Git para administrar el desarrollo de flujos de trabajo en varios entornos de vRealize Orchestrator.

---

**Nota** Si desea usar Git para administrar el inventario de objetos, la implementación de vRealize Orchestrator debe utilizar una licencia de vRealize Automation. Para obtener más información, consulte *Habilitación de funciones de vRealize Orchestrator con licencias en Instalación y configuración de vRealize Orchestrator*.

---

Ahora puede insertar objetos en las ramas y extraer objetos de ellas. A través de la ramificación, puede administrar el desarrollo de grupos específicos de objetos de vRealize Orchestrator antes de que se vuelvan a combinar en la rama principal.

En este caso práctico, se utiliza un proyecto de GitLab para administrar objetos de vRealize Orchestrator que usan el tiempo de ejecución de Python. Este caso práctico representa un ejemplo de la funcionalidad de Git en vRealize Orchestrator y no representa los límites del ámbito de las funciones.

---

**Nota** Si ya conoce GitHub, puede utilizar un repositorio de GitHub en este caso práctico.

---

## Procedimiento

### 1 Preparar su entorno de GitLab

Cree una rama de Git para los objetos de Python de vRealize Orchestrator.

### 2 Configurar una conexión a un repositorio de Git

Como **administrador**, puede configurar una conexión entre la implementación de vRealize Orchestrator y un proyecto o repositorio de Git.

### 3 Insertar cambios en un repositorio de Git

Inserte los cambios en los objetos locales de vRealize Orchestrator en su repositorio de Git integrado. En este caso práctico, vamos a insertar cambios en una acción de vRealize Orchestrator basada en Python para una rama de Git específica.

## Preparar su entorno de GitLab

Cree una rama de Git para los objetos de Python de vRealize Orchestrator.

### Requisitos previos

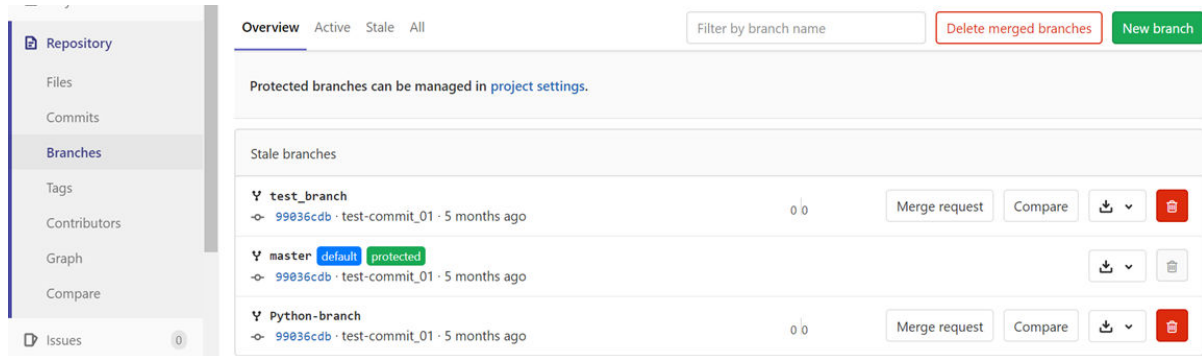
Cree un proyecto de GitLab para su entorno de vRealize Orchestrator. Consulte [Crear un proyecto](#).

### Procedimiento

- 1 Inicie sesión en su cuenta de GitLab.
- 2 Vaya hasta el proyecto de GitLab.
- 3 En el panel de navegación de la izquierda, seleccione **Repositorio > Ramas**.

- 4 En la pestaña **Descripción**, haga clic en **Nueva rama**.
- 5 En **Nombre de rama**, introduzca **Python-branch**.
- 6 Deje la opción **Crear desde** como **master**.
- 7 Haga clic en **Crear rama**.

Ha creado una rama para los objetos de vRealize Orchestrator basados en Python.



## Configurar una conexión a un repositorio de Git

Como **administrador**, puede configurar una conexión entre la implementación de vRealize Orchestrator y un proyecto o repositorio de Git.

Si desea usar Git para administrar el inventario de objetos de vRealize Orchestrator, debe configurar una conexión con el repositorio de Git mediante vRealize Orchestrator Client.

**Nota** No puede agregar varios repositorios de Git de diferentes cuentas a través de SSH porque vRealize Orchestrator crea una clave SSH para cada instancia. Para agregar varios repositorios de Git, puede agregarlos a través de HTTP como se describe en esta documentación.

### Requisitos previos

- Compruebe que el entorno de vRealize Orchestrator utiliza una licencia de vRealize Automation.
- Genere un token de acceso para el proyecto de GitLab y cópielo en el portapapeles para poder usarlo durante el proceso de configuración. Consulte [Crear un token de acceso personal](#).

**Nota** En este caso práctico, está utilizando un proyecto de GitLab. Si ya conoce GitHub, puede utilizar un repositorio de GitHub. Para obtener información sobre cómo generar un token de GitHub, consulte [Crear un token de acceso personal para la línea de comandos](#).

### Procedimiento

- 1 Inicie sesión en el vRealize Orchestrator Client como **administrador**.
- 2 Vaya a **Administración > Repositorios de Git**.
- 3 Haga clic en **Agregar repositorio**.

- 4 Introduzca la dirección URL de su repositorio de Git.

Por ejemplo, <https://gitlab.com/minombredeusuario/my-vro-repo>.

---

**Nota** También puede establecer una conexión con el protocolo SSH.

---

- 5 Introduzca el nombre de usuario de su perfil de Git.
- 6 Introduzca el token de acceso de su repositorio de Git.
- 7 Para validar la conexión con el repositorio de Git, haga clic en **Validar**.
- 8 (opcional) Cambie el nombre que utilizó para identificar el repositorio en el vRealize Orchestrator Client.
- 9 (opcional) Agregue una breve descripción para el repositorio de Git conectado.
- 10 Para activar el repositorio de Git conectado, haga clic en **Establecer como repositorio activo**.

---

**Nota** Solo puede haber un repositorio de Git activo a la vez. Puede cambiar el repositorio de Git activo en la página **Repositorios de Git**.

---

- 11 Seleccione la rama en la que desea insertar los cambios. En este caso práctico, está utilizando **Python-branch**. Consulte [Preparar su entorno de GitLab](#).

---

**Nota** Puede cambiar la rama de Git seleccionada siempre que desee después de finalizar la configuración inicial de Git.

---

- 12 Para finalizar el proceso de configuración, haga clic en **Guardar**.

#### Pasos siguientes

Vuelva al menú **Repositorios de Git** y confirme que el estado del repositorio es **Activo**.

## Insertar cambios en un repositorio de Git

Inserte los cambios en los objetos locales de vRealize Orchestrator en su repositorio de Git integrado. En este caso práctico, vamos a insertar cambios en una acción de vRealize Orchestrator basada en Python para una rama de Git específica.

Puede insertar un conjunto de cambios locales en un repositorio de Git. Cada conjunto de cambios puede constar de uno o varios objetos de vRealize Orchestrator modificados.

---

**Nota** El proceso de inserción y descarte de conjuntos de cambios en un repositorio de Git no está limitado por los permisos de grupo. Por lo tanto, un desarrollador de flujos de trabajo de un grupo puede insertar o descartar cambios locales que haya realizado otro desarrollador.

---

#### Requisitos previos

- Compruebe que haya creado una rama de Git. Consulte [Preparar su entorno de GitLab](#).
- Compruebe que haya configurado una conexión con un repositorio de Git. Consulte [Configurar una conexión a un repositorio de Git](#).

- Compruebe que la integración de Git esté configurada para insertar cambios en la rama **Python-branch** de Git.
- Cree un objeto de vRealize Orchestrator basado en Python. Por ejemplo, consulte [Integrar Amazon Web Services en vRealize Orchestrator mediante Python](#).

## Procedimiento

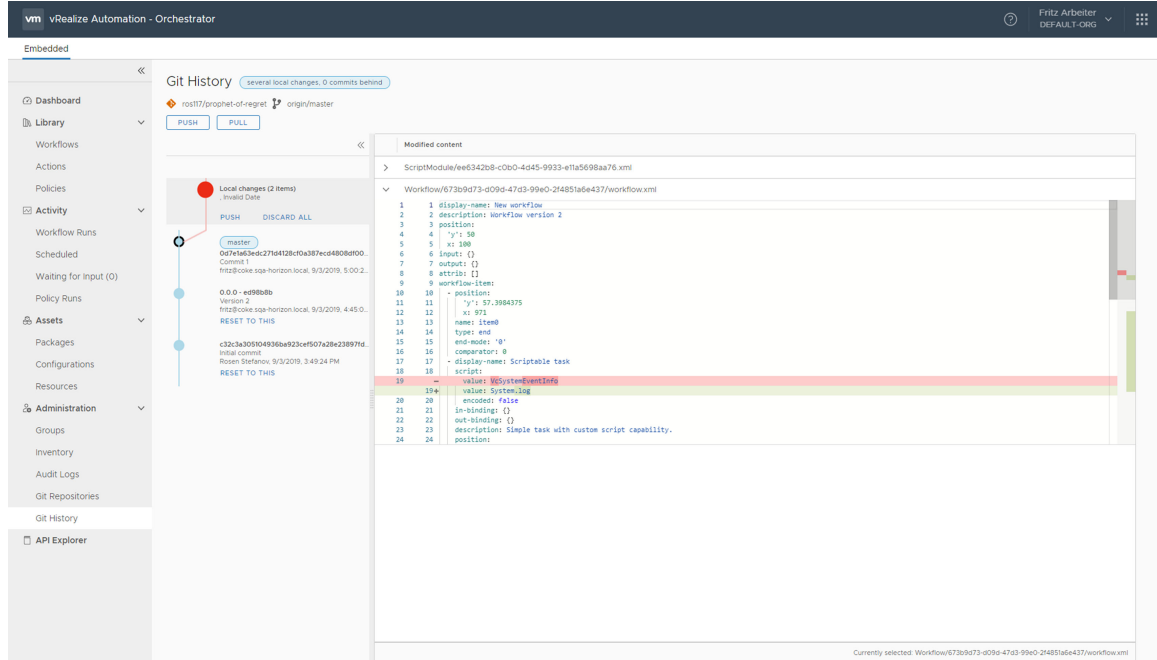
- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 Edite la acción de Python.
  - a Desplácese hasta **Biblioteca > Acciones** y seleccione la acción de Python.
  - b Realice algunos cambios secundarios en la acción, como cambiar la descripción.
  - c Guarde la acción.
- 3 Inserte los cambios en el repositorio de Git.

**Nota** Los cambios locales también se pueden insertar en cada nivel de objeto; para ello, hay que hacer clic en la opción **Versión** que se muestra en la parte inferior del editor de objetos.

- a Vaya a **Administración > Historial de Git**.

**Historial de Git** muestra las diferencias actuales entre la rama de la versión local y la rama seleccionada del repositorio de Git. Puede expandir la entrada de cualquier objeto de vRealize Orchestrator modificado para ver las diferencias entre las versiones.

**Nota** Puede descartar un conjunto de cambios locales si selecciona **Descartar todo**.



- b Haga clic en **Insertar**.



- c Escriba un título para la confirmación.
  - d (opcional) Introduzca una descripción breve para la confirmación.
  - e Seleccione los cambios de la acción de Python que desea insertar en el repositorio de Git.
- 4 Para finalizar la inserción del conjunto de cambios locales en el repositorio de Git, haga clic en **Insertar**.

#### Pasos siguientes

Después de finalizar el desarrollo en la rama de Git, combínela con la rama principal. Consulte [Cómo crear una solicitud de combinación](#).

## Usar módulos de terceros para llamar a la API de proyecto de vRealize Automation

En este caso práctico de vRealize Orchestrator se demuestra cómo se puede llamar a la API de proyecto de vRealize Automation mediante módulos de terceros.

Puede utilizar los siguientes tiempos de ejecución en sus scripts de acción y flujo de trabajo:

- Python 3.7
- Node.js 14
- PowerCLI 11/Powershell 6.2
- PowerCLI 12.3.0/PowerShell 7.1

---

**Nota** El tiempo de ejecución de PowerCLI incluye PowerShell y los siguientes módulos: VMware.PowerCLI, PowerNSX y PowervRA.

---

En este caso práctico se obtiene información sobre cómo crear acciones de vRealize Orchestrator que usan módulos de dependencia de terceros para conectarse a la API de proyecto de vRealize Automation.

---

**Importante** Antes de comenzar a desarrollar el script personalizado, asegúrese de que está familiarizado con los conceptos básicos del uso de scripts de Python, Node.js y PowerShell en vRealize Orchestrator. Consulte [Conceptos básicos de los scripts de Python, Node.js y PowerShell](#).

---

## Crear un script de Python que llame a la API de proyecto de vRealize Automation

Cree un script de ejemplo que utilice Python para llamar a la API de proyecto de vRealize Automation.

## Requisitos previos

Compruebe que se instaló Python 3 y el instalador del paquete del índice de paquetes de Python (Python Package Index, PIP). Consulte la [página de descargas de Python](#) y el [índice de paquetes de Python](#).

## Procedimiento

- 1 En el equipo local, abra un shell de línea de comandos.
- 2 Cree una carpeta `vro-python-vra`.

```
mkdir vro-python-vra
```

- 3 Desplácese hasta la carpeta `vro-python-vra`.

```
cd vro-python-vra
```

- 4 Cree un script de Python llamado `handler.py`.

```
touch handler.py
```

El script `handler.py` debe definir una función que acepte dos argumentos: el contexto de la ejecución del flujo de trabajo de vRealize Orchestrator y las entradas enlazadas de vRealize Orchestrator.

```
def handler(context, inputs):  
    print('Hello, your inputs were ' + inputs)  
    return None
```

---

**Nota** Mediante las bibliotecas de registro estándar, todo lo que se registre en la acción que utiliza el script también se muestra en el registro del flujo de trabajo. Las entradas y los valores devueltos del script deben tener configurados en el cliente de vRealize Orchestrator los parámetros de entrada y los tipos de valores devueltos correspondientes. Por ejemplo, la entrada `vRAUrl` en el script debe tener un parámetro de entrada correspondiente denominado `vRAUrl` en el cliente de vRealize Orchestrator. De forma similar, si el script devuelve un valor de cadena, el tipo de valor devuelto configurado en el cliente de vRealize Orchestrator también debe ser una cadena. Si la acción devuelve un objeto complejo, puede utilizar un tipo de valor devuelto `Properties` o `Composite Type`.

---

## 5 Instale el módulo de solicitudes de Python.

**Importante** Los módulos de dependencia de terceros deben instalarse en una carpeta de nivel raíz en la carpeta de scripts `vro-python-vra` principal. En este caso práctico se crea una carpeta `lib` para el módulo de solicitudes.

- a Cree una carpeta `lib`.

```
mkdir lib
```

- b Instale el módulo de solicitudes.

```
pip3 install requests -t lib/
```

## 6 Agregue el módulo de solicitudes al script `handler.py`.

```
import requests

def handler(context, inputs):
    print('Hello, your inputs were ' + inputs)
    return None
```

## 7 Cree una solicitud GET a la API de proyecto de vRealize Automation.

```
token = ''
vRAUrl = ''
r = requests.get(vRAUrl + '/iaas/api/projects', headers={'Authorization': 'Bearer ' + token})

print('Got response ' + r.text)
```

**8** Defina los valores de `token` y de `vRAUrl` .

- a Recupere el token de acceso mediante la API del servicio de identidad de vRealize Automation. Consulte [Obtener el token de acceso para la API de vRealize Automation](#).
- b Para el valor de `vRAUrl`, defina el script para que utilice un parámetro de entrada de vRealize Orchestrator con el mismo nombre.

```
vRAUrl = inputs["vRAUrl"]
```

- c Agregue los nuevos valores al archivo `handler.py`.

```
import requests

def handler(context, inputs):
    token = 'ACCESS_TOKEN'
    vRAUrl = inputs["vRAUrl"]

    r = requests.get(vRAUrl + '/iaas/api/projects', headers={'Authorization': 'Bearer ' + token})

    print('Got response ' + r.text)

    return r.json()
```

**Nota** Dado que la respuesta de la API de proyecto de vRealize Automation se devuelve en formato JSON, utilice un tipo de valor devuelto `Properties` o `Composite Type` para la acción de vRealize Orchestrator.

- 9 Cree un paquete ZIP que contenga el archivo `handler.py` y la carpeta `lib` del módulo de solicitudes.

```
zip -r --exclude=*.zip -X vro-python-vra.zip .
```

**Pasos siguientes**

Importe el script de PowerShell en una acción de vRealize Orchestrator. Consulte [Crear acciones en el cliente de vRealize Orchestrator](#).

## Crear un script de Node.js que llame a la API de proyecto de vRealize Automation

Cree un script de ejemplo que utilice Node.js para llamar a la API de proyecto de vRealize Automation.

**Requisitos previos**

Descargue e instale Node.js 14. Consulte la [página de descargas de Node.js](#).

**Procedimiento**

- 1 En el equipo local, abra un shell de línea de comandos.

**2** Cree una carpeta `vro-node-vra`.

```
mkdir vro-node-vra
```

**3** Desplácese hasta la carpeta `vro-node-vra`.

```
cd vro-node-vra
```

**4** Cree un script de Node.js llamado `handler.js`.

```
touch handler.js
```

El script `handler.js` debe definir una función que acepte dos argumentos: el contexto de la ejecución del flujo de trabajo de vRealize Orchestrator y las entradas enlazadas de vRealize Orchestrator.

```
exports.handler = (context, inputs) => {
  console.log('Hello, your inputs were ' + inputs);
  return null;
}
```

**Nota** Mediante las bibliotecas de registro estándar, todo lo que se registre en la acción que utiliza el script también se muestra en el registro del flujo de trabajo. Las entradas y los valores devueltos del script deben tener configurados en el cliente de vRealize Orchestrator los parámetros de entrada y los tipos de valores devueltos correspondientes. Por ejemplo, la entrada `vRAUrl` en el script debe tener un parámetro de entrada correspondiente denominado `vRAUrl` en el cliente de vRealize Orchestrator. De forma similar, si el script devuelve un valor de cadena, el tipo de valor devuelto configurado en el cliente de vRealize Orchestrator también debe ser una cadena. Si la acción devuelve un objeto complejo, puede utilizar un tipo de valor devuelto `Properties` o `Composite Type`.

**5** Instale el módulo de solicitudes de Node.js.

```
npm install request
```

**Importante** Los módulos de dependencia de terceros deben instalarse en la carpeta `node_modules` de nivel raíz en la carpeta de scripts `vro-node-vra` principal. No mueva ni cambie el nombre de esta carpeta.

**6** Agregue el módulo de solicitudes al script `handler.js`.

```
const request = require('request');

exports.handler = (context, inputs) => {
  console.log('Hello, your inputs were ' + inputs);
  return null;
}
```

## 7 Cree una solicitud GET a la API de proyecto de vRealize Automation.

```
const token = '';
const vRAUrl = '';
request.get(vRAUrl + '/iaas/api/projects', { 'auth': { 'bearer': token } }, function
(error, response, body) {
    console.log('Got response ' + body);
});
```

## 8 Defina los valores de `token` y de `vRAUrl`.

- Recupere el token de acceso mediante la API del servicio de identidad de vRealize Automation. Consulte [Obtener el token de acceso para la API de vRealize Automation](#).
- Para el valor de `vRAUrl`, defina el script para que utilice un parámetro de entrada de vRealize Orchestrator con el mismo nombre.

```
const vRAUrl = inputs.vRAUrl;
```

- Agregue los nuevos valores al archivo `handler.js`.

```
const request = require('request');
exports.handler = (context, inputs, callback) => {
    const vRAUrl = inputs.vRAUrl;
    const token = 'ACCESS_TOKEN';
    request.get(vRAUrl + '/iaas/api/projects', { 'auth': { 'bearer': token } },
function (error, response, body) {
    console.log('Got response ' + body);
    callback(null, JSON.parse(body));
});
}
```

**Nota** Dado que la respuesta de la API de proyecto de vRealize Automation se devuelve en formato JSON, utilice un tipo de valor devuelto `Properties` o `Composite Type` para la acción de vRealize Orchestrator.

## 9 Cree un paquete ZIP que contenga el archivo `handler.js` y la carpeta `node_modules` del módulo de solicitudes.

```
zip -r --exclude=*.zip -X vro-node-vra.zip .
```

### Pasos siguientes

Importe el script de Node.js en una acción de vRealize Orchestrator. Consulte [Crear acciones en el cliente de vRealize Orchestrator](#).

## Crear un script de PowerShell que llame a la API de proyecto de vRealize Automation

Cree un script de ejemplo que utilice PowerShell para llamar a la API de proyecto de vRealize Automation.

**Procedimiento**

1 En el equipo local, abra un shell de línea de comandos.

2 Cree una carpeta `vro-powershell-vra`.

```
mkdir vro-powershell-vra
```

3 Desplácese hasta la carpeta `vro-powershell-vra`.

```
cd vro-powershell-vra
```

4 Cree un script de PowerShell llamado `handler.ps1`.

```
touch handler.ps1
```

El script `handler.ps1` debe definir una función que acepte dos argumentos: el contexto de la ejecución del flujo de trabajo de vRealize Orchestrator y las entradas enlazadas de vRealize Orchestrator.

```
function Handler {
    Param($context, $inputs)

    $inputsString = $inputs | ConvertTo-Json -Compress
    Write-Host "Inputs were $inputsString"
}
```

---

**Nota** Mediante las bibliotecas de registro estándar, todo lo que se registre en la acción que utiliza el script también se muestra en el registro del flujo de trabajo. Las entradas y los valores devueltos del script deben tener configurados en el cliente de vRealize Orchestrator los parámetros de entrada y los tipos de valores devueltos correspondientes. Por ejemplo, la entrada `vRAUrl` en el script debe tener un parámetro de entrada correspondiente denominado `vRAUrl` en el cliente de vRealize Orchestrator. De forma similar, si el script devuelve un valor de cadena, el tipo de valor devuelto configurado en el cliente de vRealize Orchestrator también debe ser una cadena. Si la acción devuelve un objeto complejo, puede utilizar un tipo de valor devuelto `Properties` o `Composite Type`.

---

## 5 Instale el módulo de aserciones de PowerShell.

**Importante** Los módulos de dependencia de terceros deben instalarse en una carpeta de nivel raíz en la carpeta de scripts `vro-powershell-vra` principal. En este caso práctico se crea una carpeta `Modules` para el módulo de aserciones.

- a Cree una carpeta `Modules`.

```
mkdir Modules
```

- b Instale el módulo de aserciones.

```
pwsh -c "Save-Module -Name Assert -Path ./Modules/ -Repository PSGallery"
```

## 6 Agregue el módulo de aserciones al script `handler.ps1`.

```
Import-Module Assert

function Handler {
    Param($context, $inputs)

    $inputsString = $inputs | ConvertTo-Json -Compress
    Write-Host "Inputs were $inputsString"
}
```

## 7 Cree una solicitud GET a la API de proyecto de vRealize Automation que utiliza el cmdlet `Invoke-RestMethod`.

```
$token = ''
$vRAUrl = ''
$projectsUrl = $vRAUrl + "/project-service/api/projects"
$response = Invoke-RestMethod $projectsUrl + '/iaas/api/projects' -Headers
@{'Authorization' = "Bearer $token"} -Method 'GET'

Write-Host "Got response: $response"
```

## 8 Defina los valores de `token` y de `vRAUrl`.

- a Recupere el token de acceso mediante la API del servicio de identidad de vRealize Automation. Consulte [Obtener el token de acceso para la API de vRealize Automation](#).
- b Agregue los atributos del módulo de aserciones `Assert-NotNull` y `Assert-Type`.

```
$token | Assert-NotNull
$token | Assert-Type String
```



- c Para el valor de `vRAUrl`, defina el script para que utilice un parámetro de entrada de vRealize Orchestrator con el mismo nombre.

```
$vRAUrl = $inputs.vRAUrl
```

- d Agregue los nuevos valores al archivo `handler.ps1`.

```
Import-Module Assert
$ErrorActionPreference = "Stop"
function Handler {
    Param($context, $inputs)
    $token = "ACCESS_TOKEN"
    $token | Assert-NotNull
    $token | Assert-Type String
    $vRAUrl = $inputs.vRAUrl
    $projectsUrl = $vRAUrl + "/project-service/api/projects"
    $response = Invoke-RestMethod $projectsUrl -Headers @{'Authorization' = "Bearer $token"} -Method 'GET'

    Write-Host "Got response: $response"

    return $response
}
```

**Nota** Dado que la respuesta de la API de proyecto de vRealize Automation se devuelve en formato JSON, utilice un tipo de valor devuelto `Properties` o `Composite Type` para la acción de vRealize Orchestrator.

- 9 Cree un paquete ZIP que contenga el archivo `handler.ps1` y la carpeta `Modules` del módulo de aserciones.

```
zip -r --exclude=*.zip -X vro-powershell-vra.zip .
```

### Pasos siguientes

Importe el script de PowerShell en una acción de vRealize Orchestrator. Consulte [Crear acciones en el cliente de vRealize Orchestrator](#).

# Administrar flujos de trabajo

# 5

Un flujo de trabajo es una serie de acciones y decisiones que se ejecutan de forma secuencial. vRealize Orchestrator proporciona una biblioteca de flujos de trabajo que llevan a cabo tareas de administración comunes. vRealize Orchestrator también proporciona bibliotecas de las acciones individuales que realizan los flujos de trabajo.

Los flujos de trabajo combinan acciones, decisiones y resultados que, cuando se llevan a cabo en un determinado orden, concluyen una tarea o un proceso específico en un entorno virtual. Los flujos de trabajo realizan tareas como el aprovisionamiento de máquinas virtuales, la copia de seguridad, el mantenimiento regular, el envío de correo electrónico, las operaciones de SSH, la administración de la infraestructura física y otras operaciones de utilidad general. Las entradas aceptan los flujos de trabajo según su función. Puede crear flujos de trabajo que se ejecuten de acuerdo con programas definidos o que se ejecuten si se producen ciertos eventos anticipados. La información la puede proporcionar el usuario, otros usuarios, otro flujo de trabajo o acción, o bien un proceso externo, como una llamada de servicio web de una aplicación. Los flujos de trabajo validan y filtran la información en cierta medida antes de su ejecución.

Los flujos de trabajo pueden llamar a otros flujos de trabajo. Por ejemplo, puede tener un flujo de trabajo que llame a otro flujo de trabajo para crear una máquina virtual nueva.

Los flujos de trabajo se crean utilizando el entorno de desarrollo integrado (Integrated Development Environment, IDE) de la interfaz del vRealize Orchestrator Client, que proporciona acceso a la biblioteca de flujos de trabajo y permite ejecutar flujos de trabajo en el motor de flujos de trabajo. El motor de flujos de trabajo también puede tomar objetos de bibliotecas externas que se conectan a vRealize Orchestrator. Esta función permite personalizar procesos o implementar funciones proporcionadas por aplicaciones de terceros.

Este capítulo incluye los siguientes temas:

- [Flujos de trabajo estándar en la biblioteca de flujos de trabajo de vRealize Orchestrator](#)
- [Crear flujos de trabajo en el cliente de vRealize Orchestrator](#)
- [Editar flujos de trabajo y acciones desde el flujo de trabajo principal](#)
- [Diseñador de formularios de entrada de vRealize Orchestrator](#)
- [Solicitudes de interacción del usuario en el cliente de vRealize Orchestrator](#)
- [Programar flujos de trabajo en el cliente de vRealize Orchestrator](#)
- [Buscar referencias de objetos en flujos de trabajo](#)

## Flujos de trabajo estándar en la biblioteca de flujos de trabajo de vRealize Orchestrator

vRealize Orchestrator proporciona una biblioteca estándar de flujos de trabajo que puede usar para automatizar las operaciones de la infraestructura virtual. Los flujos de trabajo de la biblioteca estándar están bloqueados como solo lectura. Para personalizar un flujo de trabajo estándar, debe duplicar ese flujo de trabajo. Los flujos de trabajo duplicados o personalizados que cree son completamente editables.

Se puede acceder al contenido de la biblioteca de flujos de trabajo a través del menú **Biblioteca > Flujos de trabajo** de la instancia de vRealize Orchestrator Client basada en HTML5. Los flujos de trabajo estándar y personalizados del cliente se organizan mediante etiquetas. Así, por ejemplo, puede acceder al flujo de trabajo **Generar par de claves** escribiendo **SSH** en el cuadro de búsqueda de la biblioteca de flujos de trabajo.

---

**Nota** No se pueden agregar nuevas etiquetas a los flujos de trabajo estándar, a menos que duplique el flujo de trabajo.

---

## Crear flujos de trabajo en el cliente de vRealize Orchestrator

Puede utilizar el vRealize Orchestrator Client para crear y editar flujos de trabajo.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Seleccione **Biblioteca > Flujos de trabajo**.
- 3 Haga clic en **Nuevo flujo de trabajo**.
- 4 Introduzca el nombre del nuevo flujo de trabajo y haga clic en **Crear**.
- 5 Utilice el editor de flujos de trabajo para configurar las variables, las entradas y las salidas del flujo de trabajo, la estructura del esquema y la presentación del flujo de trabajo.
- 6 Para terminar de editar el flujo de trabajo, haga clic en **Guardar**.

---

**Nota** Para hacer un seguimiento de los cambios efectuados en los flujos de trabajo, consulte la pestaña **Historial de versiones**. Para obtener más información, consulte [Historial de versiones de objetos de vRealize Orchestrator](#).

---

### Pasos siguientes

Puede utilizar la función de reproducción de tokens de vRealize Orchestrator para optimizar el rendimiento de los flujos de trabajo. Para obtener más información, consulte [Usar la reproducción de tokens de flujos de trabajo en el cliente de vRealize Orchestrator](#).

## Editar flujos de trabajo y acciones desde el flujo de trabajo principal

Edite flujos de trabajo y acciones directamente desde el flujo de trabajo principal en vRealize Orchestrator Client.

La edición de acciones y flujos de trabajo secundarios directamente desde el flujo de trabajo principal puede ayudar a simplificar el desarrollo de flujos de trabajo.

### Requisitos previos

Cree un flujo de trabajo que llame a otro flujo de trabajo, una acción o ambos.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Biblioteca > Flujos de trabajo** y seleccione su flujo de trabajo.
- 3 Seleccione la pestaña **Esquema**.
- 4 En función del tipo de objeto, haga doble clic en el **Elemento Flujo de trabajo** o el **Elemento de acción** en el lienzo del flujo de trabajo.
- 5 Edite el objeto.
- 6 Para finalizar la edición de la acción o el flujo de trabajo secundario, haga clic en **Guardar**.
- 7 Para volver al flujo de trabajo principal, cierre el editor de objetos.

## Diseñador de formularios de entrada de vRealize Orchestrator

Si un flujo de trabajo precisa de parámetros de entrada, se abre un cuadro de diálogo en el que los usuarios proporcionan los valores necesarios. El contenido, el diseño y la presentación de este cuadro de diálogo se pueden organizar con el diseñador de formularios de entrada.

El diseñador de formularios de entrada se encuentra en la pestaña **Formulario de entrada** del editor de flujos de trabajo. Este diseñador consta de un menú de navegación, un lienzo de diseño y un menú de propiedades. Puede arrastrar entradas y elementos genéricos del menú de la izquierda al lienzo de diseño. En el lienzo, puede establecer la posición de los parámetros de entrada, organizarlos en pestañas de entrada independientes y configurar las propiedades de los parámetros de entrada.

---

**Nota** En el diseñador de formularios no se puede usar contenido de la pestaña **Variables** del editor de flujos de trabajo. Solo se pueden usar parámetros de la pestaña **Entrada/salida**.

---

### Elementos genéricos

Puede agregar elementos genéricos (como menús desplegables y cuadros de texto de contraseña) al diseñador de formularios de entrada. Los elementos genéricos no se

corresponden con los parámetros de entrada reales, pero se pueden enlazar a parámetros de entrada.

## Crear el cuadro de diálogo de parámetros de entrada del flujo de trabajo en el cliente de vRealize Orchestrator

Puede utilizar el diseñador de formularios de entrada para crear y personalizar el cuadro de diálogo de parámetros de entrada del flujo de trabajo.

### Requisitos previos

Compruebe que el flujo de trabajo tenga una lista definida de parámetros de entrada.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Vaya a **Biblioteca > Flujos de trabajo**.
- 3 Seleccione su flujo de trabajo personalizado.
- 4 Haga clic en la pestaña **Formulario de entrada**.
- 5 (opcional) Cree pestañas para usarlas en el cuadro de diálogo de entrada.  
Puede usar pestañas para organizar la estructura del cuadro de diálogo.
- 6 Seleccione los parámetros de entrada.
- 7 Edite las propiedades de los parámetros de entrada.  
Para obtener más información sobre las propiedades de los parámetros de entrada, consulte [Propiedades de parámetros de entrada en el cliente de vRealize Orchestrator](#).
- 8 (opcional) Añada elementos genéricos al lienzo y vincúlelos con los parámetros de entrada.
- 9 (opcional) Agregue la validación externa a los parámetros de entrada. Para obtener más información, consulte [Usar acciones para validar entradas de flujos de trabajo de vRealize Orchestrator](#).
- 10 Haga clic en **Guardar**.

### Resultados

Ha creado el diseño del cuadro de diálogo del flujo de trabajo y ha establecido las propiedades de los parámetros de entrada.

## Propiedades de parámetros de entrada en el cliente de vRealize Orchestrator

Puede definir las propiedades de los parámetros para restringir los parámetros de entrada que los usuarios proporcionan cuando ejecutan flujos de trabajo de vRealize Orchestrator.

Con vRealize Orchestrator, puede definir las propiedades de parámetros que se utilizan para cuantificar los valores de los parámetros de entrada empleados en los flujos de trabajo. Las propiedades de parámetros definidas imponen límites en los tipos y los valores de los parámetros de entrada que los usuarios proporcionan en los flujos de trabajo de vRealize Orchestrator.

Las propiedades de parámetros validan los parámetros de entrada y modifican la presentación de los cuadros de texto que aparecen en el cuadro de diálogo de los parámetros de entrada. Algunas propiedades de parámetro pueden crear dependencias entre parámetros.

| Propiedad de parámetro                 | Descripción  |
|--|--|
| <b>Etiqueta</b>                        | Establezca la etiqueta del parámetro de entrada.   |
| <b>Tipo de visualización</b>           | Establezca el tipo de visualización del cuadro de texto de entrada.  |
| <b>Visibilidad</b>                     | Establezca la visibilidad del parámetro de entrada.  |
| <b>Solo lectura</b>                    | Configure el cuadro de texto de entrada como de solo lectura.  |
| <b>Ayuda personalizada</b>             | Establezca la descripción del indicador de entrada del parámetro de entrada.   |
| <b>Valor predeterminado</b>            | Establezca el valor predeterminado del parámetro de entrada.   |
| <b>Paso</b>                            | Se usa para entradas de tipo de número. Se establece en función de cuánto aumenta el valor del parámetro de entrada por clic.  |
| <b>Obligatorio</b>                     | Establece si el valor del parámetro de entrada es obligatorio o no.  |
| <b>Expresión regular</b>               | Valida la entrada utilizando una expresión regular.  |
| <b>Valor mínimo</b>                    | Establezca el valor o la longitud mínimos del parámetro.   |
| <b>Valor máximo</b>                    | Establezca el valor o la longitud máximos del parámetro.   |
| <b>Cuadro de texto de coincidencia</b> | Establezca el valor del parámetro de entrada para que coincida con el valor de otro parámetro de entrada.  |
| <b>Origen de valor</b>                 | Establezca el origen del valor de las propiedades del parámetro en las pestañas <b>Aspecto</b> , <b>Valor</b> y <b>Restricciones</b> .<br><br><b>Nota</b> Puede importar el valor de las acciones externas mediante el uso de <b>Origen externo</b> . Las acciones disponibles se filtran por tipo de parámetro. |

## Usar acciones para validar entradas de flujos de trabajo de vRealize Orchestrator

Utilice acciones externas para validar las entradas de sus flujos de trabajo personalizados.

## Requisitos previos

Cree un flujo de trabajo personalizado con parámetros de entrada. Para obtener más información, consulte [Crear flujos de trabajo en el cliente de vRealize Orchestrator](#).

Puede utilizar el diseñador de formularios de entrada para crear validaciones externas de las entradas de flujo de trabajo. Las validaciones externas utilizan scripts de acción que devuelven un valor de cadena cuando el valor del parámetro de entrada contiene un error. Si el valor del parámetro de entrada es válido, la validación externa no devuelve nada.

## Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Cree una acción de validación.
  - a Vaya a **Biblioteca > Acciones**.
  - b Haga clic en **Nueva acción**.
  - c Introduzca la información necesaria en la pestaña **Resumen**.
  - d Introduzca los parámetros de entrada de la acción de validación.

---

**Nota** Los nombres de los parámetros de entrada de la acción de validación deben ser idénticos a los nombres de los parámetros de entrada del flujo de trabajo que se van a validar.

---

- e Introduzca el script de la acción de validación en la pestaña **Script**.

```
if (in_1=="invalid") {
    return "in_1 can't be invalid!";
}

if (in_2=="invalid") {
    return "in_2 can't be invalid!";
}

//inputs are valid, return nothing
```

---

**Nota** El script anterior es solo un ejemplo, y no representa el alcance completo de los scripts de validación que se pueden utilizar.

---

- f Haga clic en **Guardar**.
- 3 Aplique la validación externa.
  - a Vaya a **Biblioteca > Flujos de trabajo**.
  - b Seleccione su flujo de trabajo personalizado.
  - c Seleccione la pestaña **Formulario de entrada**.
  - d Seleccione el icono del portapapeles situado en la parte superior izquierda de la pantalla.
  - e Arrastre un elemento de validación de vRealize Orchestrator al lienzo.

- f Seleccione el elemento de validación, introduzca una etiqueta de validación y seleccione la acción de validación.
  - g (opcional) Cree elementos de validación adicionales.
  - h Haga clic en **Guardar**.
- 4 Ejecute el flujo de trabajo.

Si la validación detecta un error, devolverá una cadena. Si la validación es correcta, la validación no devolverá nada y la ejecución del flujo de trabajo continuará.

#### Resultados

Ha creado una validación externa para su flujo de trabajo personalizado de vRealize Orchestrator.

## Solicitudes de interacción del usuario en el cliente de vRealize Orchestrator

Los flujos de trabajo pueden solicitar al usuario datos adicionales antes de que se finalicen.

Los flujos de trabajo que requieren la interacción del usuario suspenden sus operaciones hasta que el usuario proporciona los parámetros de entrada solicitados. Los flujos de trabajo definen qué usuarios pueden proporcionar la información solicitada y envían las solicitudes de interacción según corresponda. Los flujos de trabajo que están a la espera de entradas de usuario se muestran en el panel **Ejecuciones de flujos de trabajo recientes** del panel de control de vRealize Orchestrator Client y en el menú de notificaciones de la parte superior derecha.

## Programar flujos de trabajo en el cliente de vRealize Orchestrator

Puede usar una programación para automatizar las ejecuciones de flujo de trabajo de vRealize Orchestrator.

Para programar ejecuciones de flujo de trabajo, se establece la fecha, la hora y los intervalos con que se ejecutará la tarea programada.

#### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Seleccione el flujo de trabajo en el menú **Biblioteca** y, en el panel de flujo de trabajo, haga clic en **Programar**.
- 3 Configure los parámetros de la tarea programada en las categorías **General**, **Programación** y **Flujo de trabajo**.

---

**Nota** La categoría de parámetro **Flujo de trabajo** solo está visible en los flujos de trabajo que requieren parámetros de entrada.

---



| Parámetro             | Descripción   |
|-----------------------|---|
| Nombre                | El nombre de la tarea programada.   |
| Descripción           | Una descripción breve que detalla el propósito de la tarea programada.  |
| Inicio                | Fecha y hora de la primera ejecución programada del flujo de trabajo.   |
| Iniciar si ya pasó    | Seleccione si desea que se inicie el flujo de trabajo si la hora programada ya ha pasado. <b>Sí</b> inicia el flujo de trabajo programado inmediatamente. <b>No</b> inicia el flujo de trabajo en la próxima repetición programada. |
| Programar             | Establezca el patrón de periodicidad y las entradas de desencadenadores de eventos de la tarea programada.  |
| Fecha de finalización | Solo se puede ver si está seleccionada la opción <b>Sin periodicidad</b> . Fecha y hora en que finaliza la tarea programada.  |
| Flujo de trabajo      | Introduzca los parámetros de entrada del flujo de trabajo.  |

4 Haga clic en **Crear**.

#### Resultados

Ha creado una tarea programada para el flujo de trabajo. Los flujos de trabajo programados se muestran en **Actividad > Programado**. Para eliminar tareas programadas, haga clic en **Eliminar** en el panel de programación.

## Editar una tarea programada en el cliente de vRealize Orchestrator

Las tareas programadas se pueden editar para cambiar parámetros como fecha, hora y periodicidad del flujo de trabajo programado.

#### Requisitos previos

Cree una tarea de flujo de trabajo programada.

#### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Seleccione la tarea programada en **Actividad > Programado**.
- 3 Haga clic en **Editar** en el panel de control del flujo de trabajo.
- 4 Edite la programación y haga clic en **Guardar**.

**Nota** Los parámetros de entrada establecidos al crear la tarea programada son de solo lectura y no se pueden editar. Para cambiar estos parámetros, cree una nueva tarea programada para este flujo de trabajo.

## Buscar referencias de objetos en flujos de trabajo

Como desarrollador de flujos de trabajo, puede utilizar la información de referencia de objetos para optimizar el ciclo de vida de desarrollo.

Con vRealize Orchestrator Client, puede encontrar información de referencia de objetos. Esta función tiene dos funciones:

- **Buscar dependencias:** busque información sobre las dependencias de objetos en los flujos de trabajo. Las dependencias pueden incluir otros flujos de trabajo, acciones, elementos de recursos y elementos de configuración.
- **Buscar usos:** conozca si el flujo de trabajo seleccionado se utiliza en otros flujos de trabajo de la biblioteca de vRealize Orchestrator Client.

Puede acceder a la información sobre referencias de objetos desde el editor de flujos de trabajo o desde la biblioteca de vRealize Orchestrator Client en la vista de tarjetas, la vista de lista o la vista de árbol. Para obtener más información sobre los diferentes tipos de organización de contenido de la biblioteca de vRealize Orchestrator Client, consulte [Organización de contenido en vRealize Orchestrator Client](#).

El siguiente procedimiento muestra cómo se puede acceder a las referencias de objeto desde el editor de flujos de trabajo.

### Requisitos previos

Desarrolle un flujo de trabajo que incluya al menos una referencia de objeto.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Biblioteca > Flujos de trabajo** y seleccione su flujo de trabajo.
- 3 Para obtener información sobre las dependencias de objetos, haga clic en **Buscar dependencias**.

---

**Nota** En la ventana emergente de dependencias, puede seleccionar los objetos a los que se hace referencia en la lista. Al seleccionar un objeto, se abre una pestaña de vRealize Orchestrator Client independiente donde se pueden ver los detalles del objeto seleccionado, o editarlo.

---

- 4 Para obtener información sobre dónde se utiliza el flujo de trabajo seleccionado, haga clic en **Buscar usos**.

# Administrar acciones

# 6

Para modificar los flujos de trabajo de vRealize Orchestrator, agregue scripts de acciones.

El vRealize Orchestrator Client proporciona bibliotecas de acciones predefinidas y un editor de acciones para scripts de acción personalizados. Las acciones representan funciones individuales que se utilizan como bloques de creación en flujos de trabajo.

Las acciones son funciones de JavaScript. Las acciones admiten múltiples parámetros de entrada y tienen un solo valor de devolución. Las acciones pueden llamar a cualquier objeto en la API de vRealize Orchestrator o a objetos en cualquier API que se pueda importar en vRealize Orchestrator mediante un complemento.

Cuando se ejecuta un flujo de trabajo, una acción toma parámetros de entrada de las variables del flujo de trabajo. Estas variables pueden ser parámetros de entrada iniciales del flujo de trabajo o bien variables establecidas por otros elementos del flujo de trabajo cuando se ejecutan.

El editor de acciones incluye una función de autocompletar para scripts, y un explorador de API que cuenta con los tipos de creación de scripts disponibles y su documentación.

Este capítulo incluye los siguientes temas:

- [Crear acciones en el cliente de vRealize Orchestrator](#)
- [Ejecutar y depurar acciones](#)
- [Conceptos básicos de los scripts de Python, Node.js y PowerShell](#)
- [Límites de tiempo de ejecución de los scripts de Python, Node.js y PowerShell](#)

## Crear acciones en el cliente de vRealize Orchestrator

Puede utilizar el vRealize Orchestrator Client para crear, editar y eliminar scripts de acción.

Puede utilizar los siguientes tiempos de ejecución al crear acciones:

- Python 3.7
- Node.js 14
- PowerCLI 11/Powershell 6.2

- PowerCLI 12.3.0/PowerShell 7.1

---

**Nota** El tiempo de ejecución de PowerCLI incluye PowerShell y los siguientes módulos: VMware.PowerCLI, PowerNSX y PowervRA.

---

### Requisitos previos

Antes de crear un script de PowerShell, Node.js o Python, asegúrese de que está familiarizado con los conceptos básicos para desarrollar scripts compatibles con vRealize Orchestrator que usen estos tiempos de ejecución. Consulte [Conceptos básicos de los scripts de Python, Node.js y PowerShell](#).

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Vaya a **Biblioteca > Acciones**.
- 3 Haga clic en **Nueva acción**.
- 4 En la pestaña **General**, escriba el nombre y el nombre de módulo de la acción.

---

**Nota** El nombre y el nombre de módulo deben ser únicos para cada acción. El nombre de la acción debe ser una función JavaScript válida. El nombre de la acción debe ser una única palabra que solamente puede contener letras, números y los símbolos de dólar ("\$\$") y guion bajo ("\_"). El nombre de módulo debe constar de palabras separadas por un carácter de punto (".").

---

- 5 (opcional) Cree una descripción, un número de versión, etiquetas y permisos de grupo para la acción.
- 6 En la pestaña **Script**, agregue entradas de acción, seleccione el tipo de retorno de la salida y escriba el script.

---

**Nota** Al seleccionar **ZIP** del menú desplegable **Tipo**, puede importar un origen de scripts externo y, si corresponde, sus módulos de dependencia.

---

- 7 Para finalizar la edición de la acción, haga clic en **Guardar**.

Aparecerá un mensaje para indicar que la acción se ha guardado.

### Pasos siguientes

Para ver un ejemplo de caso práctico de cómo se pueden utilizar acciones de vRealize Orchestrator, consulte [Integrar Amazon Web Services en vRealize Orchestrator mediante Python](#).

## Ejecutar y depurar acciones

Si desea mejorar las acciones, puede ejecutarlas y depurarlas directamente desde el editor de acciones.

Se pueden ejecutar y depurar acciones directamente desde el editor de acciones del cliente de vRealize Orchestrator. Con esta función, puede garantizar que las acciones se llevarán a cabo según lo previsto cuando se integren en los flujos de trabajo.

## Ejecutar acciones en el cliente de vRealize Orchestrator

Como diseñador de flujos de trabajo, desea ejecutar las acciones antes de integrarlas en un flujo de trabajo.

### Requisitos previos

Cree una acción. Consulte [Crear acciones en el cliente de vRealize Orchestrator](#).

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Vaya a **Biblioteca > Acciones** y seleccione la acción que desea ejecutar.
- 3 Haga clic en **Ejecutar**.
- 4 Introduzca los parámetros de entrada que se requieren y haga clic en **Ejecutar**.

Una vez finalizada la ejecución de la acción, haga clic en la pestaña **Resultados/Entradas**. Si se produce un error en la ejecución de la acción, este se mostrará en rojo en esta pestaña. Podrá ver los detalles de la ejecución de la acción en el elemento **Resultados de acción**.

---

**Nota** Los resultados de la ejecución de la acción no se guardarán.

---

## Depurar acciones en el cliente de vRealize Orchestrator

Como diseñador de flujos de trabajo, puede depurar acciones mediante la inserción de puntos de interrupción en el script.

vRealize Orchestrator incluye una herramienta de depuración integrada que se puede utilizar para depurar el script y las propiedades de entrada de la acción. El proceso de depuración se puede iniciar en el editor de acciones. Para ello, debe insertar puntos de interrupción en las líneas del script de la acción.

---

**Nota** La herramienta de depuración integrada solo funciona con acciones que usan el tiempo de ejecución predeterminado de JavaScript. Para ver un ejemplo de cómo se depuran los scripts de acción que utilizan diferentes tiempos de ejecución, consulte [Depurar la acción de Amazon Web Services](#).

---

### Requisitos previos

Cree una acción. Consulte [Crear acciones en el cliente de vRealize Orchestrator](#).

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Vaya a **Biblioteca > Acciones** y seleccione la acción que desea depurar.

- 3 En el editor de acciones, agregue puntos de interrupción a las líneas del script de acción que desea depurar.

- 4 Haga clic en **Depurar**.

- 5 Introduzca los parámetros de entrada de la acción y haga clic en **Ejecutar**.

Comenzará la ejecución de una acción en el modo de depuración.

- 6 Cuando la ejecución de la acción esté en pausa tras llegar a un punto de interrupción, seleccione una de las opciones disponibles:

| Opción                  | Descripción   |
|-------------------------|---|
| Continuar               | Reanuda la ejecución de la acción hasta que se alcanza otro punto de interrupción o finaliza la ejecución de la acción.   |
| Entrar                  | Entra en la función de acción actual. Si el depurador no puede avanzar en la línea actual de la función, realizará una operación de <b>Paso a paso por procedimientos</b> . |
| Ignorar                 | El depurador continuará en la siguiente línea de la función actual.   |
| Valor devuelto del paso | El depurador entra en la línea que se realizará cuando se devuelva la función actual.   |

- 7 (opcional) En la pestaña **Depurador**, agregue expresiones.

- 8 (opcional) En la pestaña **Depurador**, edite el valor de las variables.

## Conceptos básicos de los scripts de Python, Node.js y PowerShell

Al crear un script para usarlo en vRealize Orchestrator, debe comprobar que este tenga la estructura y el formato correctos.

### Tiempos de ejecución compatibles

Para desarrollar acciones y flujos de trabajo de vRealize Orchestrator, puede utilizar los siguientes tiempos de ejecución:

- Python 3.7
- Node.js 14
- PowerCLI 11/Powershell 6.2
- PowerCLI 12.3.0/PowerShell 7.1

**Nota** El tiempo de ejecución de PowerCLI incluye PowerShell y los siguientes módulos: VMware.PowerCLI, PowerNSX y PowervRA.

Puede agregar cualquier código fuente personalizado a los nuevos tiempos de ejecución, pero debe seguir el formato funcional correcto para aceptar el contexto y las entradas, así como para devolver un resultado procedente o destinado al motor de vRealize Orchestrator.

## Recomendaciones de creación de scripts

Para las tareas de creación de scripts más sencillas, puede agregar elementos de **Tarea de scripts** al esquema de flujo de trabajo. Puede utilizar acciones de vRealize Orchestrator para realizar tareas de scripts más complejas.

El uso de acciones proporciona dos ventajas específicas:

- Las acciones se pueden crear, actualizar, importar y exportar independientemente de los flujos de trabajo.
- Las acciones son objetos independientes que pueden ejecutarse y depurarse en su propio entorno, lo que puede llevar a un proceso de desarrollo más ágil. Consulte [Ejecutar y depurar acciones](#).

## Requisitos de la función de script

El nombre predeterminado de la función de script es **handler**. La función acepta dos argumentos: contexto y entrada. El contexto es un objeto de asignación que contiene información del sistema. Por ejemplo, `vroURL` puede contener la dirección URL de la instancia de vRealize Orchestrator a la que se desea llamar, mientras que `executionId` contiene el identificador de token de una ejecución de flujo de trabajo.

Una entrada es un objeto de asignación que contiene todas las entradas que se proporcionan a las acciones. Por ejemplo, si se define una entrada en una acción llamada `myInput`, puede accederse a ella desde el argumento de entradas (como `inputs.myInput` o `inputs["myInput"]`) en función del tiempo de ejecución. Cualquier elemento que se devuelva de la función será el resultado de la acción. Por tanto, el tipo de valor devuelto de la acción debe corresponder al tipo de contenido que el script devuelve en vRealize Orchestrator. Si se devuelve un número primitivo, el tipo de valor devuelto de la acción debe ser un número. Si se devuelve una cadena, el tipo de valor devuelto de la acción debe ser una cadena. Si se devuelve un objeto complejo, el tipo de valor devuelto debe asignarse a `Properties` o a `Composite Type`. Estos mismos principios también se aplican a las matrices.

Estos son los tipos de parámetros de entrada y salida admitidos para los tiempos de ejecución de Python, Node.js y PowerShell:

- `String`
- `Number`
- `Boolean`
- `Date`
- `Properties`
- `Composite Type`

## Definir el controlador de entrada

De forma predeterminada, el valor del controlador de entrada es `handler.handler`. Este valor significa que el motor de vRealize Orchestrator busca un archivo de nivel superior en el paquete ZIP llamado `handler.py`, `handler.js` o `handler.ps1`, el cual que incluye una función denominada `handler`. Cualquier diferencia entre los nombres de la función y el archivo del controlador debe reflejarse en el valor del controlador de entrada. Por ejemplo, si el controlador principal se denomina `index.js` y la función se llama `callMe`, debe establecer el valor del controlador de entrada como `index.callMe`.

## Depurar scripts de tiempo de ejecución en un IDE externo

vRealize Orchestrator admite la depuración de scripts de Python y Node.js en un IDE externo. Los scripts de PowerShell no se pueden depurar en un IDE externo.

## Límites de tiempo de ejecución de los scripts de Python, Node.js y PowerShell

Puede que algunos scripts de Python, Node.js o PowerShell requieran que se cambien los valores de memoria y tiempo de espera en el vRealize Orchestrator Client.

El vRealize Orchestrator Client utiliza un conjunto de valores predeterminados de memoria y tiempo de espera para los scripts de acción de Python, Node.js y PowerShell:

- Memoria: 64 MB
- Tiempo de espera: 180 segundos

Si el script de acción supera uno de estos valores predeterminados (o ambos), se produce un error en la ejecución de la acción. Por ejemplo, puede que el script de acción utilice varios módulos de dependencia de terceros. En tal caso, es posible que el límite de memoria predeterminado de 64 MB no sea suficiente.

Para evitar que se produzcan errores al ejecutar las acciones debido a recursos insuficientes, cambie los valores de memoria y tiempo de espera en el editor de acciones.

---

**Nota** También puede considerar la posibilidad de dividir el script en varios elementos de tareas de scripts, los cuales pueden agregarse a los flujos de trabajo.

---

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 Desplácese hasta **Biblioteca > Acciones** y seleccione la acción.
- 3 Seleccione la pestaña **Script**.
- 4 En **Límites de tiempo de ejecución**, cambie los valores de memoria y tiempo de espera.
- 5 Haga clic en **Guardar**.



- 6 Para probar los nuevos límites de tiempo de ejecución, haga clic en **Depurar**.

# Administrar elementos de configuración

# 7

Un elemento de configuración es una lista de variables que puede utilizar para configurar constantes en toda una implementación del servidor de vRealize Orchestrator.

Puede utilizar elementos de configuración con el fin de que las variables estén disponibles para todos los flujos de trabajo, las acciones y las políticas que se ejecutan en el servidor de vRealize Orchestrator.

Si crea un paquete que contiene un flujo de trabajo, una acción o una política que utiliza una variable de un elemento de configuración, vRealize Orchestrator incluye automáticamente el elemento de configuración en el paquete. Si importa un paquete que contiene un elemento de configuración a otro servidor de vRealize Orchestrator, también puede importar los valores de variable del elemento de configuración. Por ejemplo, si crea un flujo de trabajo que requiere valores de variable que dependen del servidor de vRealize Orchestrator en el que se ejecuta, la configuración de dichas variables en un elemento de configuración le permite exportar ese flujo de trabajo para que otro servidor de vRealize Orchestrator pueda utilizarlo. Por lo tanto, los elementos de configuración permiten intercambiar flujos de trabajo, acciones y políticas entre servidores con mayor facilidad.

---

**Nota** No es posible importar valores de una variable de elemento de configuración a partir de un elemento de configuración exportado de vRealize Orchestrator 5.1 o una versión anterior.

---

Este capítulo incluye los siguientes temas:

- [Crear elementos de configuración en el cliente de vRealize Orchestrator](#)

## Crear elementos de configuración en el cliente de vRealize Orchestrator

Con los elementos de configuración, puede establecer variables comunes en un servidor de vRealize Orchestrator. Todos los elementos que se ejecutan en el servidor pueden usar las variables establecidas en un elemento de configuración.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Activos > Configuraciones**.

- 3 Seleccione **Nueva configuración**.
- 4 Escriba el nombre del elemento de configuración.
- 5 Seleccione la pestaña **Variables**.
- 6 Para crear una variable local, haga clic en **Nueva**.
  - a Introduzca el nombre de la variable.
  - b Seleccione el tipo de variable.

---

**Nota** Para crear una matriz de variables de configuración, active la casilla de verificación **Matriz**.

---

- c (opcional) Introduzca un valor para la variable de configuración.
  - d Haga clic en **Guardar**.
- 7 Para finalizar la creación de un elemento de configuración, haga clic en **Guardar**.

#### Pasos siguientes

Puede utilizar el elemento de configuración para proporcionar variables para flujos de trabajo, acciones o políticas.

# Administrar políticas

# 8

Las políticas son activadores de eventos que supervisan la actividad del sistema. Las políticas responden a eventos predefinidos que emiten los cambios en el estado o el rendimiento de objetos específicos de vRealize Orchestrator.

Las políticas son una serie de reglas, medidores, umbrales y filtros de eventos que ejecutan determinados flujos de trabajo o scripts cuando se producen ciertos eventos predefinidos en vRealize Orchestrator o en las tecnologías a las que accede vRealize Orchestrator a través de los complementos. vRealize Orchestrator evalúa constantemente las reglas de la política mientras esta se ejecuta. Por ejemplo, puede implementar medidores y umbrales de política que supervisen el comportamiento de los objetos de vCenter Server de los tipos `VC:HostSystem` y `VC:VirtualMachine`.

Este capítulo incluye los siguientes temas:

- [Crear y aplicar políticas en el cliente de vRealize Orchestrator](#)
- [Elementos de política en el cliente de vRealize Orchestrator](#)
- [Administrar ejecuciones de políticas en el cliente de vRealize Orchestrator](#)

## Crear y aplicar políticas en el cliente de vRealize Orchestrator

Puede utilizar políticas a fin de supervisar la actividad del sistema de vRealize Orchestrator para determinados eventos.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Biblioteca > Políticas**.
- 3 Seleccione **Nueva política**.  
Ha creado una política en blanco.
- 4 Introduzca un nombre y un número de versión de política.
- 5 Seleccione la pestaña **Variables**.

6 Para crear una variable local, haga clic en **Nueva**.

- a Introduzca el nombre de la variable.
- b Seleccione el tipo de variable.

---

**Nota** Para crear una matriz de variables de política, active la casilla de verificación **Matriz**.

---

- c Introduzca el valor de variable.

---

**Nota** Para importar el valor de una variable de elemento de configuración, puede usar la opción **Enlazar a configuración**.

---

- d Haga clic en **Guardar**.

7 En la pestaña **Definición**, agregue elementos de política y establezca controladores de eventos.

Para obtener más información sobre elementos de política, consulte [Elementos de política en el cliente de vRealize Orchestrator](#).

8 Haga clic en **Guardar**.

Ha configurado la política.

#### Pasos siguientes

Para iniciar una política, selecciónela y haga clic en **Ejecutar**. Introduzca el nombre de ejecución de la política y, si se le solicita, los parámetros de entrada necesarios.

Para ver el estado de la política, desplácese hasta **Actividad > Ejecuciones de políticas**.

## Elementos de política en el cliente de vRealize Orchestrator

Puede utilizar elementos de política para que se ejecuten flujos de trabajo o scripts de vRealize Orchestrator predefinidos cuando se produzca un evento.

Puede agregar un elemento de política para activar las ejecuciones de flujos de trabajo o scripts como respuesta a los eventos activados por los objetos. Con el elemento de evento periódico, puede programar ejecuciones de scripts o flujos de trabajo. Con el elemento raíz, puede establecer el comportamiento de inicio o de detención de las políticas. Los elementos de política pueden tener controladores de eventos que definen cuándo se deben ejecutar los elementos de la política.

---

**Nota** Los controladores de eventos que activan elementos de política pueden ser flujos de trabajo o scripts de acción. Si agrega un flujo de trabajo y un script a un controlador de eventos, la política omitirá el activador del script y solo utilizará el activador del flujo de trabajo.

---

| Controlador de eventos | Descripción  |
|------------------------|--|
| <b>OnInit</b>          | El elemento de política se activa cada vez que se inicia la política.  |
| <b>OnExit</b>          | El elemento de política se activa cada vez que se detiene la política.   |
| <b>OnExecute</b>       | Utilizado por el elemento de evento periódico. Activa el elemento de política durante el tiempo especificado en el elemento de evento periódico. |

**Nota** Las tecnologías conectadas a la base de datos de vRealize Orchestrator pueden poseer controladores de eventos únicos. Por ejemplo, el complemento SNMP permite utilizar el controlador de eventos **OnTrap** al crear elementos basados en políticas de SNMP.

Los elementos de política se configuran en la pestaña **Definición** de la ventana de edición de políticas.

## Administrar ejecuciones de políticas en el cliente de vRealize Orchestrator

Puede utilizar vRealize Orchestrator Client para administrar la prioridad de las políticas y el comportamiento de inicio del servidor de las políticas para cuando se reinicie el servidor de vRealize Orchestrator.

### Requisitos previos

Cree y ejecute una política. Para obtener más información, consulte [Crear y aplicar políticas en el cliente de vRealize Orchestrator](#).

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 2 Desplácese hasta **Actividad > Ejecuciones de políticas**.
- 3 Haga clic en la ejecución de política que desee administrar.
- 4 Haga clic en **Detener**.  
El estado de la política cambiará a **Detenida**.
- 5 En la pestaña **General**, establezca la prioridad de la política y el comportamiento de inicio del servidor.
- 6 Para reiniciar la política, haga clic en **Ejecutar**.  
El estado de la política cambiará a **En ejecución**.

# Administrar elementos de recursos

## 9

Los flujos de trabajo pueden utilizar objetos que se crean de forma independiente de vRealize Orchestrator como atributos. Para usar objetos externos como atributos en flujos de trabajo, impórtelos en el servidor como elementos de recursos.

Los objetos que los flujos de trabajo de vRealize Orchestrator pueden usar como elementos de recursos son, entre otros, archivos de imagen, scripts, plantillas XML o archivos HTML. Los flujos de trabajo que se ejecutan en el servidor de vRealize Orchestrator pueden utilizar cualquier elemento de recursos que se importe en vRealize Orchestrator.

Si se importa un objeto en vRealize Orchestrator como elemento de recursos, es posible efectuar cambios en el objeto en una sola ubicación y propagarlos automáticamente a todos los flujos de trabajo que usan ese elemento de recursos.

Un elemento de recursos tiene un tamaño máximo de 16 MB.

Puede importar, exportar, restaurar, actualizar y eliminar un elemento de recurso.

# Administrar paquetes

# 10

Use vRealize Orchestrator Client para crear, exportar e importar paquetes. Los paquetes pueden utilizarse para exportar objetos de flujo de trabajo y usarlos en otras instancias de vRealize Orchestrator.

Los paquetes pueden contener flujos de trabajo, acciones, políticas, elementos de configuración o elementos de recursos.

Al agregar un elemento a un paquete, vRealize Orchestrator comprueba las dependencias y agrega cualquier elemento dependiente al paquete. Por ejemplo, si agrega un flujo de trabajo que utiliza otros flujos de trabajo o acciones, vRealize Orchestrator agrega esos flujos de trabajo y esas acciones al paquete.

Al importar un paquete, el servidor compara las versiones de los diferentes elementos de su contenido con los elementos locales correspondientes. La comparación muestra las diferencias de versiones entre los elementos locales y los importados. El usuario puede decidir si se importa el paquete; asimismo, puede seleccionar la importación de determinados elementos.

En el caso de la mayoría de los objetos creados en el vRealize Orchestrator Client, los paquetes son la única forma de exportar e importar estos objetos, aparte de los elementos de recursos.

Los paquetes utilizan la administración de derechos digitales para controlar el modo en el que el servidor destinatario puede utilizar el contenido del paquete. vRealize Orchestrator firma paquetes y los cifra para la protección de datos. Los paquetes pueden hacer el seguimiento de los usuarios que exportan y redistribuyen elementos mediante el uso de certificados X509.

## Crear un paquete en el cliente de vRealize Orchestrator

Puede exportar e importar flujos de trabajo, políticas, acciones, referencias de complementos, elementos de recursos y elementos de configuración en paquetes. Todos los elementos dependientes que estén relacionados con objetos del paquete se agregan al paquete automáticamente con el fin de garantizar la compatibilidad entre versiones. Para eliminar elementos dependientes, primero deberá eliminar el objeto de paquete relacionado.

En el caso de la mayoría de los objetos creados en el vRealize Orchestrator Client, los paquetes son la única forma de exportar e importar estos objetos, aparte de los elementos de recursos.



### Requisitos previos

Compruebe que el servidor de vRealize Orchestrator contenga objetos como flujos de trabajo, acciones y políticas que se puedan agregar a un paquete.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Activos > Paquetes**.
- 3 Haga clic en **Nuevo paquete**.
- 4 En la pestaña **General**, añada un nombre y una descripción para el paquete.

---

**Nota** No puede usar caracteres especiales al asignar nombres a los paquetes del vRealize Orchestrator Client.

---

- 5 En la pestaña **Contenido**, haga clic en **Agregar**.
- 6 Seleccione los objetos que desea agregar al paquete y haga clic en **Agregar**.

---

**Nota** Los elementos dependientes se agregan automáticamente al paquete, pero no se muestran en la pestaña **Contenido** durante la creación del paquete. Para ver los elementos dependientes, seleccione la pestaña **Contenido** después de crear el paquete.

---

- 7 Para finalizar la creación del paquete, haga clic en **Crear**.

## Exportar un paquete en el cliente de vRealize Orchestrator

Puede utilizar el vRealize Orchestrator Client para exportar paquetes a otro entorno de vRealize Orchestrator.

### Requisitos previos

Cree un paquete que contenga los objetos de vRealize Orchestrator que desea exportar. Para obtener más información, consulte [Crear un paquete en el cliente de vRealize Orchestrator](#).

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Activos > Paquetes**.
- 3 Haga clic en **Exportar** en el paquete.

#### 4 (opcional) Seleccione otras opciones de exportación.

| Opción  | Descripción   |
|---|---|
| Agregar valores de atributos de configuración al paquete              | Exporta los valores de atributo de los elementos de configuración.            |
| Agregar valores del atributo SecureString de configuración al paquete | Exporta los valores del atributo de configuración <code>SecureString</code> . |
| Agregar etiquetas globales al paquete                                 | Exporta las etiquetas globales.   |

#### 5 Establezca los derechos de acceso para los usuarios que importen el paquete.

| Opción             | Descripción  |
|--------------------|--|
| Ver contenido      | El usuario puede ver el contenido del paquete.                             |
| Agregar al paquete | El usuario puede agregar contenido del paquete importado a otros paquetes. |
| Editar contenido   | El usuario puede editar el contenido del paquete.                          |

#### 6 Haga clic en **Aceptar**.

**Nota** Los archivos que tengan la extensión `.package` se guardarán en una carpeta predeterminada del equipo local. Para definir una carpeta personalizada, puede cambiar la configuración de almacenamiento del explorador.

### Resultados

Ha exportado el paquete. Ahora puede utilizar los objetos exportados en otro entorno de vRealize Orchestrator.

## Importar un paquete en el cliente de vRealize Orchestrator

Utilice la instancia de vRealize Orchestrator Client para importar paquetes de flujos de trabajo. Al importar los paquetes, puede reutilizar objetos procedentes de un servidor de vRealize Orchestrator en otro servidor.

### Requisitos previos

- Cree una copia de seguridad de los objetos de vRealize Orchestrator estándar que haya modificado.
- En el servidor remoto, cree y exporte un paquete con los objetos que desea importar.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Activos > Paquetes**.
- 3 Haga clic en **Importar**, busque el archivo `.package` que desea importar y haga clic en **Abrir**.

**4** Revise la información del paquete importado.

- a La pestaña **General** contiene información sobre el paquete importado, como el nombre, la descripción, el número de elementos que contiene y el certificado.

Es posible que deba indicar que confía en el certificado del publicador de la instancia de vRealize Orchestrator de origen para poder importar el archivo.

- b En la pestaña **Elementos del paquete** se enumeran los objetos incluidos en el archivo de importación. Si la versión de un objeto en el paquete es posterior a la versión en el servidor, el sistema selecciona esa versión de objeto para importarla. Las versiones anteriores de los elementos de vRealize Orchestrator se deben seleccionar manualmente.
- c Anule la selección de **Importar los valores de atributo de la configuración** si no desea importar los valores de atributo de los elementos de configuración del paquete.
- d En el menú desplegable, seleccione si desea importar etiquetas.

**5** Haga clic en **Importar**.

# Solucionar problemas en el cliente de vRealize Orchestrator

# 11

Puede solucionar los problemas de la instancia de vRealize Orchestrator y supervisarla mediante métricas, reproducción de tokens, validación y depuración.

Este capítulo incluye los siguientes temas:

- Datos de métricas en el cliente de vRealize Orchestrator
- Usar la reproducción de tokens de flujos de trabajo en el cliente de vRealize Orchestrator
- Validar flujos de trabajo de vRealize Orchestrator
- Depurar scripts de flujos de trabajo en el cliente de vRealize Orchestrator
- Depurar flujos de trabajo por elemento de esquema
- Configurar un contenedor de Photon OS para paquetes de Python

## Datos de métricas en el cliente de vRealize Orchestrator

Los administradores de vRealize Orchestrator pueden utilizar las métricas de generación de perfiles de flujo de trabajo y del panel de control del sistema para solucionar problemas relativos al sistema y a los flujos de trabajo de vRealize Orchestrator.

La función de generación de perfiles recopila datos de métricas sobre las ejecuciones de los flujos de trabajo. La generación de perfiles de flujo de trabajo está habilitada de forma predeterminada. La generación de perfiles automática se puede deshabilitar en **Centro de control > Propiedades de extensión > profiler-8.7.0**.

El otro origen de los datos de métricas del vRealize Orchestrator Client es el panel de control del sistema, que proporciona métricas a nivel de sistema. Para obtener más información, consulte [Usar el panel de control del sistema de vRealize Orchestrator](#).

## Generar perfiles de flujos de trabajo en el cliente de vRealize Orchestrator

Puede generar perfiles de ejecuciones de flujos de trabajo para solucionar problemas y optimizar el entorno de vRealize Orchestrator.

Puede utilizar la característica de generación de perfiles de vRealize Orchestrator Client para recopilar datos de métricas sobre las ejecuciones de flujo de trabajo. Estos datos pueden ser útiles para optimizar el rendimiento de los flujos de trabajo. De forma predeterminada, las ejecuciones de flujos de trabajo se generan como perfiles automáticamente. Puede deshabilitar la generación de perfiles automática en la página **Propiedades de extensión** del centro de control de vRealize Orchestrator y ejecutar el generador de perfiles de forma manual. Para ejecutar una generación de perfiles manual, busque el flujo de trabajo en la biblioteca y seleccione **Acciones > Perfil**.

#### Requisitos previos

Ejecute un flujo de trabajo.

#### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator.
- 2 Desplácese hasta **Actividad > Ejecuciones de flujos de trabajo**.
- 3 Seleccione una ejecución de flujo de trabajo.

En el esquema de ejecución de flujo de trabajo, podrá ver datos sobre los elementos de flujo de trabajo individuales. Los datos incluyen la duración total de la ejecución, la duración máxima y el número de ejecuciones de elementos. Puede filtrar esta información en el menú desplegable situado en la parte superior derecha de la página.

- 4 Seleccione la pestaña **Rendimiento**.

Esta pestaña proporciona datos de métricas sobre el tiempo de CPU de la ejecución de flujo de trabajo, la duración de la ejecución, el tamaño de token y los datos de elementos de flujo de trabajo.

---

**Nota** Si se interrumpe la ejecución del flujo de trabajo (por ejemplo, cuando el flujo de trabajo está a la espera de más entradas), la métrica de tiempo de CPU solo captura el subproceso de tiempo de ejecución que se produjo antes de la finalización.

---

#### Pasos siguientes

Utilice los datos recopilados de la generación de perfiles para optimizar el flujo de trabajo.

## Usar el panel de control del sistema de vRealize Orchestrator

Como administrador, puede usar el panel de control del sistema de vRealize Orchestrator Client para recopilar datos de métricas útiles acerca de los nodos del entorno de vRealize Orchestrator.

Para acceder al panel de control del sistema, haga clic en la pestaña **Sistema** en la parte superior de la página del panel de control de vRealize Orchestrator Client. Los datos proporcionados incluyen:

- Estado del nodo
- Propiedades del nodo

- Configuración del clúster. La configuración del clúster solo se puede ver en el panel de control del sistema. Para cambiar esta configuración, vaya a la página **Administración de clústeres de Orchestrator** del centro de control de vRealize Orchestrator.
- Información sobre subprocessos
- Memoria de pila
- Memoria que no es de pila
- Uso del sistema de archivos
- Datos de autenticación
- Grupo de conexiones de base de datos de Orchestrator
- Argumentos de entrada de proceso

Estos datos se pueden utilizar para supervisar el estado de los nodos individuales del entorno de vRealize Orchestrator y solucionar problemas. Para desplazarse entre los nodos individuales, haga clic en la pestaña asociada a un nodo en la parte superior del panel de control del sistema.

## Usar la reproducción de tokens de flujos de trabajo en el cliente de vRealize Orchestrator

Puede usar la función de reproducción de tokens para ver las transiciones entre los elementos de las ejecuciones de flujos de trabajo.

La función de reproducción de tokens registra la información contextual de cada transición entre elementos de flujo de trabajo. Para cada elemento de flujo de trabajo, la reproducción de tokens registra cuándo se inició la ejecución del flujo de trabajo, cuándo finalizó, así como las variables que se cambiaron al final de la ejecución del elemento de flujo de trabajo. La reproducción de tokens también hace referencia a los mensajes del registro de scripts generados para cada elemento de flujo de trabajo.

---

**Nota** Los datos sobre las transiciones de los elementos del flujo de trabajo se almacenan en la base de datos PostgreSQL de vRealize Orchestrator. Estos datos se eliminan de la base de datos cuando se elimina la ejecución del flujo de trabajo.

---

### Requisitos previos

- Habilite la función de reproducción de tokens en el Centro de control.
  - a Inicie sesión en el centro de control como usuario **raíz**.
  - b Seleccione **Propiedades de extensión**.
  - c Haga clic en **tokenreplay-8.7.0**.
  - d Para habilitar la función de reproducción de tokens, haga clic en **Habilitar**.

- e Haga clic en **Guardar**.

---

**Nota** El servidor de vRealize Orchestrator puede tardar hasta 5 minutos en actualizar la extensión.

---

- Ejecute un flujo de trabajo.

---

**Nota** De forma predeterminada, la reproducción de tokens no se ejecuta automáticamente para todas las ejecuciones de flujos de trabajo en el servidor de vRealize Orchestrator. Puede ejecutar la reproducción de tokens para cada flujo de trabajo de forma individual o habilitar la extensión de reproducción de tokens para todos los flujos de trabajo desde la página **Propiedades de extensión** del Centro de control.

---

#### Procedimiento

- 1 (opcional) Habilite la reproducción de tokens para todas las ejecuciones de flujos de trabajo en el servidor de vRealize Orchestrator.

---

**Nota** Para ejecutar la reproducción de tokens individuales sin habilitar la función desde el Centro de control, haga clic en **Ejecutar con reproducción** en la página del editor de flujos de trabajo.

---

- a Inicie sesión en el centro de control como usuario **raíz**.
- b Seleccione **Propiedades de extensión**.
- c Haga clic en **tokenreplay-8.7.0**.
- d Para habilitar la función de reproducción de tokens para todos los flujos de trabajo, compruebe que la opción **Reproducción de registros para todas las ejecuciones de flujos de trabajo** esté habilitada.
- e Haga clic en **Guardar**.

---

**Nota** El servidor de vRealize Orchestrator puede tardar hasta 5 minutos en actualizar la extensión.

---

- 2 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 3 Desplácese hasta **Actividad > Ejecuciones de flujos de trabajo**.
- 4 Seleccione una ejecución de flujo de trabajo.
- 5 Seleccione un elemento de ejecución de flujo de trabajo en el menú de la izquierda.  
Las pestañas **Variable** y **Registros** ahora muestran información específica sobre ese elemento de flujo de trabajo.

## Validar flujos de trabajo de vRealize Orchestrator

vRealize Orchestrator proporciona una herramienta para la validación de flujos de trabajo. Al validar un flujo de trabajo, se pueden identificar errores en el flujo de trabajo y comprobar que los datos fluyan correctamente de un elemento al siguiente.

De forma predeterminada, vRealize Orchestrator siempre efectúa la validación de flujos de trabajo cuando ejecuta un flujo de trabajo.

Cuando valida un flujo de trabajo, la herramienta de validación crea una lista con todos los errores y advertencias. Si se hace clic en un error de la lista, el elemento del flujo de trabajo que contiene el error queda resaltado.

Si ejecuta la herramienta de validación en el editor de flujos de trabajo, la herramienta proporciona correcciones rápidas sugeridas para los errores detectados. Algunas correcciones rápidas requieren información adicional o parámetros de entrada. Otras correcciones rápidas resuelven los errores por usted.

La validación de flujos de trabajo comprueba los enlaces y las conexiones entre los elementos. La validación de flujos de trabajo no comprueba el procesamiento de datos que realiza cada elemento en el flujo de trabajo. Como resultado, un flujo de trabajo válido se puede ejecutar de manera incorrecta y producir resultados erróneos si una función de un elemento del esquema es incorrecta.

## Validar un flujo de trabajo y solucionar errores de validación en el cliente de vRealize Orchestrator

Antes de poder ejecutar un flujo de trabajo, debe validarlo. Solo puede corregir errores de validación si ha abierto el flujo de trabajo para editarlo.

### Requisitos previos

Compruebe que disponga de un flujo de trabajo completo para validar, con elementos de esquema vinculados y enlaces definidos.

### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 2 Vaya a **Biblioteca > Flujos de trabajo** y seleccione el flujo de trabajo que desea validar.
- 3 Haga clic en **Editar**.
- 4 Haga clic en **Validar** en el menú superior.

Si el flujo de trabajo es válido, aparece un mensaje de confirmación. Si el flujo de trabajo no es válido, aparece una lista de errores.



- 5 En el caso de un flujo de trabajo no válido, haga clic en el mensaje de error y siga los pasos adecuados para resolver el problema.

La herramienta de validación resalta el elemento del esquema en el que se encuentra el error añadiendo un icono rojo. Siempre que sea posible, la herramienta de validación muestra una corrección rápida.

- Si está de acuerdo con la corrección rápida propuesta, haga clic en ella para llevar a cabo la acción.
- Si no está de acuerdo con la corrección rápida propuesta, cierre el cuadro de diálogo Validación de flujos de trabajo y corrija manualmente el elemento del esquema.

---

**Importante** Compruebe siempre si la solución que propone vRealize Orchestrator es adecuada.

---

Por ejemplo, la acción propuesta puede consistir en eliminar un atributo no utilizado, cuando es posible que ese atributo no esté enlazado correctamente.

- 6 Repita los pasos anteriores hasta que haya eliminado todos los errores de validación.

#### Resultados

Ya ha validado un flujo de trabajo y ha corregido los errores de validación.

#### Pasos siguientes

Ahora puede ejecutar el flujo de trabajo.

## Depurar scripts de flujos de trabajo en el cliente de vRealize Orchestrator

Puede depurar ejecuciones de flujo de trabajo mediante la inserción de puntos de interrupción en el script de los elementos de flujo de trabajo.

Cuando se llega a un punto de interrupción, hay varias opciones para continuar el proceso de depuración. Al depurar un elemento del esquema de flujo de trabajo, puede ver información general sobre la ejecución del flujo de trabajo, modificar las variables de flujo de trabajo, agregar expresiones que se deben vigilar y ver los mensajes de registro.

---

**Nota** Realice toda la depuración de scripts en un entorno que no sea de producción.

---

#### Procedimiento

- 1 Inicie sesión en el cliente de vRealize Orchestrator como administrador.
- 2 Seleccione un flujo de trabajo de la biblioteca.
- 3 Abra el esquema de flujo de trabajo, seleccione un elemento de flujo de trabajo y haga clic en la pestaña **Creación de scripts**.

- 4 Para insertar un punto de interrupción, haga clic en el círculo rojo a la izquierda del número de línea.

---

**Nota** Solo se pueden insertar puntos de interrupción en elementos de flujo de trabajo con scripts.

---

- 5 Para ejecutar el flujo de trabajo en modo de depuración, haga clic en **Depurar**.

Si el flujo de trabajo requiere parámetros de entrada, se deben facilitar.

- 6 Cuando la ejecución del flujo de trabajo esté en pausa al llegar a un punto de interrupción, seleccione una de las opciones disponibles.

| Opción    | Descripción  |
|-----------|--|
| Continuar | Reanuda la ejecución del flujo de trabajo hasta que se alcanza otro punto de interrupción o finaliza la ejecución del flujo de trabajo.  |
| Entrar    | Puede utilizar esta opción para entrar en un elemento de flujo de trabajo. No se puede entrar en un elemento de flujo de trabajo anidado si se depura un flujo de trabajo en el Editor de flujos de trabajo. |
| Ignorar   | Omite el elemento actual del esquema y pone en pausa la ejecución del flujo de trabajo en el próximo elemento.   |

---

**Nota** Si desea indicar al depurador que ignore el punto de interrupción actual, haga clic en el punto de interrupción. Esto hace que el símbolo del punto de interrupción se reemplace por un triángulo verde.

---

- 7 (opcional) En la pestaña **Depurador**, inserte las expresiones que se deben vigilar.  
Puede utilizar expresiones para seguir la finalización de variables específicas.
- 8 (opcional) En la pestaña **Depurador**, modifique los valores de las variables.

## Depurar flujos de trabajo por elemento de esquema

Como diseñador de flujos de trabajo, puede depurar elementos de esquema individuales.

### Procedimiento

- 1 Inicie sesión en vRealize Orchestrator Client.
- 2 Desplácese hasta **Biblioteca > Flujos de trabajo** y seleccione su flujo de trabajo.
- 3 Seleccione la pestaña **Esquema**.

- 4 Seleccione el elemento de flujo de trabajo que desee depurar y haga clic en el botón de depuración situado en la parte superior izquierda del elemento.

**Nota** Al agregar un punto de interrupción a un elemento de esquema **Elemento de flujo de trabajo**, puede depurar los flujos de trabajo secundarios directamente desde el flujo de trabajo principal. Cuando el depurador llegue al elemento de esquema **Elemento de flujo de trabajo**, abra la vista de esquema del flujo de trabajo secundario.

- 5 Repítalo para los otros elementos de esquema que desee depurar.
- 6 Haga clic en **Depurar**.
- 7 Introduzca los valores de parámetros de entrada solicitados y haga clic en **Ejecutar**.

Comenzará entonces la ejecución del flujo de trabajo y se suspenderá cuando el depurador llegue a un elemento de esquema con un punto de interrupción.

- 8 Cuando se encuentre en un punto de interrupción, seleccione una de las siguientes opciones:

| Opción                  | Descripción   |
|-------------------------|---|
| Continuar               | Reanuda la ejecución del flujo de trabajo hasta que se alcanza otro punto de interrupción o finaliza la ejecución del flujo de trabajo.   |
| Entrar                  | Entre en la función de flujo de trabajo actual. Si el depurador no puede avanzar en la línea actual de la función, realizará una operación de <b>Paso a paso por procedimientos</b> . |
| Ignorar                 | El depurador continuará en la siguiente línea de la función actual.   |
| Valor devuelto del paso | El depurador entra en la línea que se realizará cuando se devuelva la función actual.   |

- 9 (opcional) En la pestaña **Variables**, edite el valor de las variables del flujo de trabajo.

The screenshot shows the VMware vRealize Orchestrator debugger interface. The top panel displays a workflow diagram with a task named "My Orchestrator Task". The middle panel shows the JavaScript code for this task, with a breakpoint set at line 1. The bottom panel shows the "Variables" tab, which lists several variables and their values.

| Variable           | Value   |
|--------------------|---------|
| i                  | Not set |
| VM                 | test    |
| workflowToLaunch   | Not set |
| workflowParameters | Not set |
| wfToken            | Not set |
| VM                 | test    |
| workflowToRun      | test    |

# Configurar un contenedor de Photon OS para paquetes de Python

Según el sistema operativo (SO) utilizado para compilar el script de Python, se pueden producir errores en los flujos de trabajo o las acciones después de importar el archivo ZIP relevante al cliente de vRealize Orchestrator.

El sistema operativo del contenedor de tiempo de ejecución utilizado para Python en vRealize Orchestrator se basa en Photon 3.0. Los paquetes de script de Python compilados para otro sistema operativo, como Linux, por ejemplo, son incompatibles con el contenedor de tiempo de ejecución. Este problema puede provocar un error en el script de Python al intentar utilizarlo como parte de los flujos de trabajo o acciones de vRealize Orchestrator. En tal caso, recibirá el siguiente mensaje de error en los registros:

```
-04:00errorCannot find module action
```

Para solucionar este problema, debe instalar el paquete de Python requerido en una carpeta de contenedor Photon OS.

## Requisitos previos

Instale Docker. Consulte el tema sobre cómo [Obtener Docker](#).

## Procedimiento

- 1 Desplácese hasta la carpeta principal del script de Python.
- 2 Cree un contenedor con la imagen base de Photon montando una carpeta de contenedor en la carpeta principal.

**Nota** El siguiente script es un comando de Docker singular que debe ejecutar en su totalidad para crear un contenedor adecuado.

```
docker run -ti -v
$(pwd) /<name_of_folder_that_contains_your_python_script>/:/
<name_of_folder_that_contains_your_python_script>
photon:3.0
```

- 3 Instale Python en el contenedor.

```
tdnf install -y python3-3.7.5-5.ph3 python3-pip-3.7.5-5.ph3
```

- 4 Desplácese hasta la carpeta de contenedor que incluye el script de Python.
- 5 Agregue el script de Python y los paquetes.

**Nota** Instale los paquetes necesarios para el script de Python en la carpeta lib.

```
pip3 install <package_name> -t lib/
```

- 6 Salga del contenedor y desplácese hasta la carpeta local montada en el contenedor.
- 7 Comprime todos los archivos y las carpetas relevantes en un archivo ZIP.
- 8 Importe el archivo ZIP al cliente de vRealize Orchestrator y valide el script ejecutándolo como parte de una acción.