

Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0

vRealize Automation 7.0

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see <http://www.vmware.com/support/pubs>.

EN-001964-02

vmware[®]

You can find the most up-to-date technical documentation on the VMware Web site at:

<http://www.vmware.com/support/>

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

Copyright © 2012–2016 VMware, Inc. All rights reserved. [Copyright and trademark information.](#)

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

| | |
|-------------------------------------------------------------------------------------------|-----------|
| Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0 | 5 |
| Updated Information | 7 |
| 1 Introduction to the VMware vRealize Orchestrator Plug-In for vRealize Automation | 9 |
| Role of vRealize Orchestrator with the vRealize Automation Plug-In | 9 |
| Installing the vRealize Automation Plug-In | 10 |
| vRealize Automation Plug-In Functional Prerequisites | 10 |
| Install the vRealize Automation Plug-In On an External vRealize Orchestrator Server | 10 |
| 2 Configuring the vRealize Automation Plug-In | 11 |
| Configuration Workflows | 11 |
| Add a vRealize Automation Host | 12 |
| Add an IaaS Host | 13 |
| 3 Using the vRealize Automation Plug-In Workflows | 15 |
| Remove Operation Restrictions | 15 |
| Restricted Operations | 16 |
| Using the vRealize Automation Plug-In Inventory | 17 |
| Using the vRealize Automation Plug-In Administration Workflows | 17 |
| Using the vRealize Automation Plug-In Infrastructure Administration Workflows | 22 |
| Create a vRealize Automation IaaS Model Entity | 24 |
| Read a vRealize Automation IaaS Model Entity | 25 |
| Using the vRealize Automation Plug-In Requests Workflows | 25 |
| Using the vRealize Automation Plug-In Sample Workflows | 26 |
| Access the vRealize Automation Plug-In API | 26 |
| 4 Example vRealize Automation Plug-In Scripts | 27 |
| CRUD Infrastructure Administration Tasks Example Scripts | 27 |
| Finding vRealize Automation Entities Example Scripts | 31 |
| Get a Resource Provisioned by vRealize Automation Example Script | 32 |
| Common Tasks Example Scripts | 33 |
| Index | 37 |

Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0

Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0 provides information and instructions about configuring and using the VMware® vRealize Orchestrator plug-in for VMware vRealize Automation 7.0.

Intended Audience

The information in *Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0* is written for experienced users who are familiar with virtual machine technology, with Orchestrator workflow development, and with VMware vRealize Automation.

For more information about Orchestrator, see

http://www.vmware.com/support/pubs/orchestrator_pubs.html.

For more information about vRealize Automation, see

<http://www.vmware.com/support/pubs/vrealize-automation.html>.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation, go to

<http://www.vmware.com/support/pubs>.

Updated Information

Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0 is updated with each release of the product or when necessary.

This table provides the update history of *Using the vRealize Orchestrator Plug-In for vRealize Automation 7.0*.

| Revision | Description |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| EN-001964-02 | <ul style="list-style-type: none">■ Updated the Business Groups and Content sections in “Using the vRealize Automation Plug-In Administration Workflows,” on page 17.■ Added information regarding roles in “Using the vRealize Automation Plug-In Administration Workflows,” on page 17.■ Added a link to the Cloud Client that you can use to import and export workflows to “Using the vRealize Automation Plug-In Administration Workflows,” on page 17.■ Updated the Infrastructure Administration and Extensibility sections in “Using the vRealize Automation Plug-In Infrastructure Administration Workflows,” on page 22.■ Updated the Requests section in “Using the vRealize Automation Plug-In Requests Workflows,” on page 25.■ Updated the Samples section in “Using the vRealize Automation Plug-In Sample Workflows,” on page 26. |
| EN-001964-01 | <ul style="list-style-type: none">■ Updated the Workflow Subscriptions section in “Using the vRealize Automation Plug-In Administration Workflows,” on page 17. |
| EN-001964-00 | Initial release. |

Introduction to the VMware vRealize Orchestrator Plug-In for vRealize Automation

1

The VMware vRealize Orchestrator plug-in for vRealize Automation allows interaction between vRealize Orchestrator and vRealize Automation.

You can use the vRealize Automation plug-in to create and run workflows for the following vRealize Automation functions:

- XaaS custom resource and blueprint management
- Catalog item and resource management and requesting
- Entitlement configuration
- Approval policy configuration
- Work item interactions
- vSphere and vCloud Director virtual machine provisioning and post-provisioning actions
- Create, read, update, and delete (CRUD) operations on the vRealize Automation IaaS model

This chapter includes the following topics:

- [“Role of vRealize Orchestrator with the vRealize Automation Plug-In,”](#) on page 9
- [“Installing the vRealize Automation Plug-In,”](#) on page 10

Role of vRealize Orchestrator with the vRealize Automation Plug-In

You use the Orchestrator client to run and create workflows and access the plug-in API. You can use either the embedded vRealize Orchestrator instance in your vRealize Automation installation, or an external vRealize Orchestrator server.

vRealize Orchestrator powers the vRealize Automation plug-in. vRealize Orchestrator is a development and process-automation platform that provides a library of extensible workflows to manage the VMware cloud stack and third party technologies.

vRealize Orchestrator allows integration with management and administration solutions through its open plug-in architecture.

Installing the vRealize Automation Plug-In

The vRealize Automation plug-in is installed on the embedded vRealize Orchestrator instance in your vRealize Automation installation. You can use the control center of an external vRealize Orchestrator server to install the vRealize Automation plug-in.

vRealize Automation Plug-In Functional Prerequisites

To install and use the vRealize Automation plug-in, your system must meet certain functional prerequisites.

vRealize Automation

You must have access to a vRealize Automation server. Version 7.0 of the plug-in works with vRealize Automation 7.0.

For information about setting up vRealize Automation, see vRealize Automation *Installing vRealize Automation 7.0*.

vRealize Orchestrator Server

Version 7.0 of the plug-in works with vRealize Orchestrator 7.0.

For information about setting up Orchestrator, see *Installing and Configuring VMware vRealize Orchestrator*.

Other Prerequisites

For full plug-in functionality, you must uninstall Web Distributed Authoring and Versioning (WebDAV) from your Microsoft Web Server (IIS) role services.

Install the vRealize Automation Plug-In On an External vRealize Orchestrator Server

You can install the plug-in on an external vRealize Orchestrator server by using the Orchestrator control center interface.

Prerequisites

- Download the `.vmoapp` file that contains the plug-in.
- Log in to the vRealize Orchestrator control center interface at `http://orchestrator_server:8283/vco-controlcenter`.

Procedure

- 1 In the Plug-Ins section, click **Manage Plug-Ins**.
- 2 Click **Browse** and locate the `.vmoapp` file.
- 3 Click **Open**.
- 4 Click **Install**.

What to do next

Configure the vRealize Automation plug-in.

Configuring the vRealize Automation Plug-In

2

You add vRealize Automation hosts and IaaS hosts to configure the plug-in.

Configuration Workflows

You can use the workflows in the **Configuration** workflow categories to manage vRealize Automation hosts.

vRealize Automation Hosts

You can access these workflows from the **Workflows** view of the Orchestrator client, in the **Configuration** subdirectory of the plug-in library.

| Workflow Name | Description |
|-----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add a vRA host | Adds a vRealize Automation host to the plug-in inventory. For tenant management and administration tasks, you can use the Inventory view to run workflows on each tenant. To use the full function of the plug-in for a tenant, create a dedicated vRealize Automation host for each tenant. |
| Add a vRA host using component registry | Adds a vRealize Automation host to the plug-in inventory with a Per User Session connection. You must be logged in to the Orchestrator client with the credentials of the vRealize Automation system administrator. To use this function with an external vRealize Orchestrator server, you must register the Orchestrator server in the vRealize Automation component registry. |
| Add the IaaS host of a vRA host | Adds the IaaS host of the selected vRealize Automation host to the plug-in inventory. |
| Remove a vRA host | Removes a vRealize Automation host from the plug-in inventory. |
| Update a vRA host | Updates a vRealize Automation host in the plug-in inventory. |
| Validate a vRA host | Validates the vRealize Automation host and the connection to it. |

NOTE If your vRealize Orchestrator server is registered in the vRealize Automation component registry, a vRealize Automation host with the name Default is automatically added. The Default host is using Per User Session connection to the default tenant. The embedded Orchestrator server in the vRealize Automation installation is registered in the vRealize Automation component registry by default.

vRealize Automation IaaS Hosts

You can access these workflows from the **Workflows** view of the Orchestrator client, in the **Infrastructure Administration > Configuration** subdirectory of the plug-in library.

The embedded vRealize Orchestrator server in the vRealize Automation installation is registered in the vRealize Automation component registry by default.

| Workflow Name | Description |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add an IaaS host | Adds a vRealize Automation IaaS host to the plug-in inventory. This workflow is functionally the same as Add the IaaS host of a vRA host, but does not require a vRealize Automation host. |
| Remove an IaaS host | Removes a vRealize Automation IaaS host from the plug-in inventory. |
| Update an IaaS host | Updates a vRealize Automation IaaS host in the plug-in inventory. |
| Validate an IaaS host | Validates the vRealize Automation IaaS host and the connection to it. |

Add a vRealize Automation Host

You can run a workflow to add a vRealize Automation host and configure the host connection parameters.

Procedure

- 1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
- 2 Click the **Workflows** view.
- 3 Select **Library > vRealize Automation > Configuration**.
- 4 Right-click the **Add a vRA host** workflow and select **Start workflow**.
- 5 Enter a unique name for the host in the **Host Name** text box.
- 6 Enter the URL address of the host in the **Host URL** text box.

For example: *https://hostname*.

- 7 Enter the name of the tenant in the **Tenant** text box.

To use the full functionality of the plug-in for a tenant, create a dedicated vRealize Automation host for each tenant.

- 8 Select whether to install the SSL certificates automatically without user confirmation.
- 9 (Optional) To configure the length of time vRealize Orchestrator waits for a connection or response from vRealize Automation, enter timeout intervals in the **Connection timeout (seconds)** and **Operation timeout (seconds)** text boxes.
- 10 Select the type of connection to the host from the **Session mode** drop-down menu.

| Option | Actions |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Shared Session | Enter the credentials for a vRealize Automation user in the Authentication username and Authentication password text boxes. |
| Per User Session | Connect using the credentials of the user that is currently logged in. You must be logged in to the Orchestrator client with the credentials of the vRealize Automation system administrator. To use this option with an external vRealize Orchestrator server, you must register the Orchestrator server in the vRealize Automation component registry. |

- 11 Click **Submit**.

What to do next

Add a vRealize Automation Infrastructure Administration host.

Add an IaaS Host

You can run a workflow to add the IaaS host of a vRealize Automation host and configure the connection parameters.

Procedure

- 1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
- 2 Click the **Workflows** view.
- 3 Select **Library > vRealize Automation > Infrastructure Administration > Configuration**.
- 4 Right-click **Add an IaaS host** and select **Start workflow**.
- 5 Select the vRealize Automation host for which you want to configure an IaaS host from the **vCAC host** drop-down menu.
- 6 Enter a unique name for the host in the **Host Name** text box.
- 7 Enter the URL of the machine on which your Model Manager is installed.
For example: `https://model_manager_machine.com`.
- 8 To install the SSL certificates, select **Yes**.
- 9 To use a proxy to access your model manager machine, select **Yes**.
If you select this option, you must provide the proxy host and the proxy port on the following page.
- 10 Click **Next**.
- 11 If you are configuring an explicit proxy, provide the proxy host URL and the port.
- 12 Click **Next**.
- 13 To configure your own timeout values, click **No**.
- 14 (Optional) To configure the length of time vRealize Orchestrator waits for a connection or response from vRealize Automation, enter timeout intervals in the **Connection timeout (seconds)** and **Operation timeout (seconds)** text boxes.
- 15 Click **Next**.
- 16 Select the host's authentication type.

| Option | Description |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SSO | Select this to use vCenter Single Sign-On. |
| NTLM | Select this to enable NT LAN Manager (NTLM) protocol-based authentication only if your Active Directory infrastructure relies on NTLM authentication. If you select this option, you must the additional NTLM credentials and authentication options. |

- 17 If you selected NTLM, click **Next** and enter the name of the Workstation machine and the NetBIOS domain name.
- 18 Click **Submit**.

Using the vRealize Automation Plug-In Workflows

3

The vRealize Automation plug-in workflow library contains workflows that you can use for common tasks such as interacting with the catalog, managing infrastructure, and creating tenants and services.

You can use custom HTTP headers, such as the vRealize Automation specific headers Tasks and Identity, and apply them in the CRUD, provisioning, and post-provisioning workflows.

This chapter includes the following topics:

- [“Remove Operation Restrictions,”](#) on page 15
- [“Using the vRealize Automation Plug-In Inventory,”](#) on page 17
- [“Using the vRealize Automation Plug-In Administration Workflows,”](#) on page 17
- [“Using the vRealize Automation Plug-In Infrastructure Administration Workflows,”](#) on page 22
- [“Using the vRealize Automation Plug-In Requests Workflows,”](#) on page 25
- [“Using the vRealize Automation Plug-In Sample Workflows,”](#) on page 26
- [“Access the vRealize Automation Plug-In API,”](#) on page 26

Remove Operation Restrictions

Some create, read, update, and delete operations are restricted beginning with version 7.0. If you used the operations in your workflows in previous versions, they will not work with 7.0 and later. You can update your workflows to the supported operations or you can re-enable the operations that you need.

To re-enable the operations, you must remove the operations that you want to enable from the operations.properties file. For a list of the operations in the file, see [“Restricted Operations,”](#) on page 16.

Procedure

- 1 From the drop-down menu in vRealize Orchestrator, select **Design**.
- 2 Click the **Resources** view.
- 3 In the resource hierarchy, select **Library > VCAC > Util**.
- 4 Create a backup and modify the operations.properties file.
 - a Right-click operations.properties and select **Save to file**.
 - b Save a copy as a backup.
 - c Create a new copy and delete the operations that you want to re-enable.
 - d Save the new file.

- 5 Replace the existing file in vRealize Orchestrator.
 - a In vRealize Orchestrator, right-click the **Util** folder and click **Import resources**.
 - b Browse to the new version of the operations.properties file and click **Open**.
 - c Click **Replace once** to save your modified version.
- 6 Restart the vRealize Orchestrator server.
- 7 Select the operations.properties file and click the **Viewer** tab.
- 8 Verify that the operations that you are enabling are no longer in the file.

The operations that you removed from the file now work in your older workflows.

What to do next

As you create new workflows, avoid using the restricted operations.

Restricted Operations

The contents of the operations.properties file contains the restricted operations. To re-enable the operation, you must remove it from the file.

The following text is the default version of the operations.properties file. To re-enable an operation, see [“Remove Operation Restrictions,”](#) on page 15.

```
#Blueprints
operation.create=ManagementModelEntities.svc@VirtualMachineTemplates
operation.update=ManagementModelEntities.svc@VirtualMachineTemplates
operation.delete=ManagementModelEntities.svc@VirtualMachineTemplates
#Blueprint properties
operation.create=ManagementModelEntities.svc@VirtualMachineProperties
operation.read=ManagementModelEntities.svc@VirtualMachineProperties
operation.update=ManagementModelEntities.svc@VirtualMachineProperties
operation.delete=ManagementModelEntities.svc@VirtualMachineProperties
#Global profiles
operation.create=ManagementModelEntities.svc@GlobalProfiles
operation.read=ManagementModelEntities.svc@GlobalProfiles
operation.update=ManagementModelEntities.svc@GlobalProfiles
operation.delete=ManagementModelEntities.svc@GlobalProfiles
#Global profile properties
operation.create=ManagementModelEntities.svc@GlobalProfileProperties
operation.read=ManagementModelEntities.svc@GlobalProfileProperties
operation.update=ManagementModelEntities.svc@GlobalProfileProperties
operation.delete=ManagementModelEntities.svc@GlobalProfileProperties
#PropertySetXml
operation.create=ManagementModelEntities.svc@PropertySetXml
operation.read=ManagementModelEntities.svc@PropertySetXml
operation.update=ManagementModelEntities.svc@PropertySetXml
operation.delete=ManagementModelEntities.svc@PropertySetXml
#Property definitions
operation.create=ManagementModelEntities.svc@PropertyDefinitions
operation.read=ManagementModelEntities.svc@PropertyDefinitions
operation.update=ManagementModelEntities.svc@PropertyDefinitions
operation.delete=ManagementModelEntities.svc@PropertyDefinitions
#Property attributes
operation.create=ManagementModelEntities.svc@PropertyAttributes
operation.read=ManagementModelEntities.svc@PropertyAttributes
```



```

operation.update=ManagementModelEntities.svc@PropertyAttributes
operation.delete=ManagementModelEntities.svc@PropertyAttributes
#Property Attribute Types
operation.create=ManagementModelEntities.svc@PropertyAttributeTypes
operation.read=ManagementModelEntities.svc@PropertyAttributeTypes
operation.update=ManagementModelEntities.svc@PropertyAttributeTypes
operation.delete=ManagementModelEntities.svc@PropertyAttributeTypes
#Control layouts
operation.create=ManagementModelEntities.svc@ControlLayouts
operation.read=ManagementModelEntities.svc@ControlLayouts
operation.update=ManagementModelEntities.svc@ControlLayouts
operation.delete=ManagementModelEntities.svc@ControlLayouts
#Amazon Virtual Machine Templates
operation.create=AmazonWSModelEntities.svc@AmazonVirtualMachineTemplates
operation.read=AmazonWSModelEntities.svc@AmazonVirtualMachineTemplates
operation.update=AmazonWSModelEntities.svc@AmazonVirtualMachineTemplates
operation.delete=AmazonWSModelEntities.svc@AmazonVirtualMachineTemplates
#Openstack Virtual Machine Templates
operation.create=OpenStackModelEntities.svc@OpenstackVirtualMachineTemplates
operation.read=OpenStackModelEntities.svc@OpenstackVirtualMachineTemplates
operation.update=OpenStackModelEntities.svc@OpenstackVirtualMachineTemplates
operation.delete=OpenStackModelEntities.svc@OpenstackVirtualMachineTemplates

```

Using the vRealize Automation Plug-In Inventory

You can use the **Inventory** view to run workflows on vRealize Automation objects.

To display the workflows that are available for an inventory object, navigate to **Tools > User preferences > Inventory** and select the **Use contextual menu in inventory** check box. After the option is enabled, when you right-click an object in the Orchestrator inventory, all available workflows for the object are displayed.

Using the vRealize Automation Plug-In Administration Workflows

You can use the administration workflows to manage vRealize Automation services, tenants, approval policies, entitlements, business groups, catalog items, and Advanced Services components.

Some of the workflows include an input parameter for the vRealize Automation host, `VCACAFE:VCACHost`. How you configured the vRealize Automation host connection determines how the roles are applied when a user runs the workflows.

- If you configured the connection as a shared session, the user account for the shared session must have the roles required to run the workflows.
- If you configured the connection as session per user, the each user who runs the workflows must have must have the required roles, just as they would in the vRealize Automation user interface.

You can find these workflows on the **Workflows** view of the Orchestrator client, in the **Administration** subdirectory of the plug-in library.

You can use the workflows in the **Approval Policies** subdirectory to create and manage approval policies.

Table 3-1. Approval Policies

| Workflow | Description |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------|
| Activate an approval policy | Activates an approval policy. After you activate an approval policy, it becomes read-only. |
| Add an approval level | Adds an always required approval level to an approval. You must select specific users and groups for the approvers. |

Table 3-1. Approval Policies (Continued)

| Workflow | Description |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Copy an approval policy | Copies an approval policy. |
| Create an approval policy | Creates a draft approval policy with no levels or approvers. To create approval levels and designate approvers for your policy, run the Add an approval level workflow. |
| Deactivate an approval policy | Deactivates an approval policy. You can also delete all existing entitlements associated with the approval policy. |
| Delete an approval policy | Deletes an approval policy that is in draft state. Active approval policies are read-only. |

You can use the workflows in the **Business Groups** subdirectory to create and manage business groups.

Table 3-2. Business Groups

| Workflow | Description |
|--------------------------|--------------------------------------------------------------------------------------------------------------------|
| Add a custom property | Adds a custom property to a business group. |
| Create a business group | Creates a business group. |
| Delete a business group | Deletes a business group. |
| Delete a custom property | Removes a custom property from a business group. |
| Update a business group | Updates details for a business group, such as default machine prefix, active directory containers, and user roles. |
| Update a custom property | Updates a custom property for a business group. |

The Administration subdirectory includes a **Business Groups (Deprecated)** subdirectory that works with versions before vRealize Automation 7.0. Use the workflows with the same name in the main folder.

You can use the workflows in the **Catalog Items** subdirectory to manage catalog items.

Table 3-3. Catalog Items

| Workflow | Description |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Activate a catalog item | Activates a catalog item. You must activate and assign a catalog item to a service before users can request it. |
| Assign a catalog item to a service | Assigns a catalog item to a service. You must activate and assign a catalog item to a service before users can request it. |
| Deactivate a catalog item | Deactivates a catalog item and removes it from the service catalog so that users cannot request it. |

You can use the workflows in the **Composite Blueprint** subdirectory to manage composite blueprints create in the design canvas.

Table 3-4. Composite Blueprint

| Workflow | Description |
|---------------------------------|------------------------------------------------------------------|
| Delete a composite blueprint | Delete an unpublished blueprint form the Design blueprints list. |
| Import a composite blueprint | Import a composite blueprint from a YAML file. |
| Publish a composite blueprint | Publish a composite blueprint that is in a draft state. |
| Unpublish a composite blueprint | Unpublish a published composite blueprint. |

You can use the workflows in the **Content** subdirectory to manage XaaS custom resources, blueprints, resource mappings, and resource actions. Some of the workflows were deprecated in 7.0. Use Cloud Client to perform the import and export actions. The Cloud Client download and documentation are available at <https://developercenter.vmware.com/tools>.

Table 3-5. Content

| Workflow | Description |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Export content (deprecated) | Export Advanced Services components as a vRealize Automation resource bundle. |
| Import content (deprecated) | Import Advanced Services components from a .zip file. You can also use the Import content workflow to transfer workflows from a .package file to a separate vRealize Orchestrator associated with your vRealize Automation. For example, if you are running an external vRealize Orchestrator and an embedded vRealize Orchestrator, and both are configured for the same vRealize Automation host, you can run the import content workflow from the external vRealize Orchestrator to install workflows on the embedded instance. |
| Transfer content (deprecated) | Transfer Advanced Services components between vRealize Automation tenants. |
| Validate content (deprecated) | Ensures that all workflows required for imported Advanced Services components exist. |

You can use the workflows in the **Entitlements** subdirectory to create and manage entitlements.

Table 3-6. Entitlements

| Workflow | Description |
|-----------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| Activate an entitlement | Activates an entitlement. |
| Assign catalog items to an entitlement | Assigns one or more catalog items to an entitlement. You can also use this workflow to assign an approval policy. |
| Assign immediate actions to an entitlement | Assigns one or more immediate actions to an entitlement. The immediate actions do not create requests. |
| Assign resource actions to an entitlement | Assigns one or more resource actions to an entitlement. You can also use this workflow to assign an approval policy. |
| Assign services to an entitlement | Assigns one or more services to an entitlement. You can also use this workflow to assign an approval policy. |
| Assign users and groups to an entitlement | Assigns one or more users or groups to an entitlement. |
| Create an entitlement (deprecated) | Creates an entitlement. Use Create and entitlement for subtenant. |
| Create an entitlement for subtenant | Creates an entitlement. |
| Deactivate an entitlement | Deactivates an entitlement. |
| Unassign users and groups from an entitlement | Remove users and groups from the list of users for an entitlement. |

You can use the workflows in the **Services** subdirectory to manage services.

Table 3-7. Services

| Workflow | Description |
|-----------------------------------|-------------------------------------------------|
| Activate a service | Activates a service. |
| Assign catalog items to a service | Assigns one or more catalog items to a service. |
| Copy a service | Copies a service. |
| Create a service | Creates a service. |
| Deactivate a service | Deactivates a service. |
| Delete a service | Deletes a service. |

You can use the workflows in the **Tenants** subdirectory to create and manage tenants.

Table 3-8. Tenants

| Workflow | Description |
|-------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add administrators | Adds one or more tenant administrators and infrastructure administrators to a tenant. |
| Add an identity store to a tenant | Adds an identity store to a tenant of a vRealize Automation host. You can run this workflow only if you are a system administrator configuring a tenant. |
| Add an identity store to a vCAC host | Adds an identity store to a tenant that is configured as a vRealize Automation host. You can run this workflow only if you are a tenant administrator configuring an identity store for your tenant. |
| Create a tenant | Creates a tenant. You must select the vRealize Automation host added with the system administrator credentials. |
| Delete an identity store from a tenant | Deletes an identity store from a tenant of a vRealize Automation host. You can run this workflow only if you are a system administrator configuring a tenant. |
| Delete an identity store from a vCAC host | Deletes an identity store from a tenant that is configured as a vRealize Automation host. You can run this workflow only if you are a tenant administrator configuring identity stores for your tenant. |
| Delete a tenant | Deletes a tenant. |
| Remove administrators | Removes one or more tenant administrators and infrastructure administrators from a tenant. |
| Update an identity store for a tenant | Updates an existing identity store for a tenant of a vRealize Automation host. You can run this workflow only if you are a system administrator configuring a tenant. |
| Update an identity store for a vCAC host | Updates an identity store for a tenant that is configured as a vRealize Automation host. You can run this workflow only if you are a tenant administrator configuring identity stores for your tenant. |
| Update a tenant | Updates the name, description, and contact email address of an existing tenant. |

You can use the workflows in the **Workflow Subscriptions** subdirectory to manage the event workflow subscriptions.

Table 3-9. Workflow Subscriptions

| Workflow | Description |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Delete a workflow subscription | Delete an unpublished workflow subscription. This workflow applies to system and tenant workflow subscriptions. |
| Export system workflow subscription | Export a system workflow subscription and save it as a vRealize Orchestrator resource element in a JSON format. A system workflow subscription is a specialized workflow subscription that reacts to system events and to events in all the tenants. |
| Export tenant workflow subscription | Export a tenant workflow subscription and save it as a resource element in a JSON format. A specialized workflow subscription that runs tenant-specific workflows. |
| Import system workflow subscription | Import a system workflow subscription from a JSON file. System workflow subscriptions are triggered for system events and can be across tenants. |
| Import tenant workflow subscription | Import an exported workflow subscription from a JSON file. These workflow subscriptions are tenant-specific. |
| Publish a workflow subscription | Publish a workflow subscription that was in a draft or unpublished state. This workflow applies to system and tenant workflow subscriptions. |
| Register a system workflow subscription | Create a system workflow subscription, including timeout and priority values. |
| Register a tenant workflow subscription | Create a tenant-specific workflow subscription, including timeout and priority values. |
| Unpublish a workflow subscription | Unpublish a published workflow subscription. This workflow applies to system and tenant workflow subscriptions. |
| Update a workflow subscription | Change the name, description, vRealize Orchestrator workflow, subscription conditions, timeout value, and priority value. You cannot update the event topic or blocking state. |

You can use the workflows in the **XaaS Custom Resources** subdirectory to create and delete XaaS custom resources.

Table 3-10. XaaS Custom Resources

| Workflow | Description |
|------------------------|----------------------------|
| Create custom resource | Creates a custom resource. |
| Delete custom resource | Removes a custom resource. |

You can use the workflows in the **XaaS Resource Actions** subdirectory to create and manage XaaS resource actions.

Table 3-11. XaaS Resource Actions

| Workflow | Description |
|------------------------|------------------------------------------------|
| Clone resource action | Creates a copy of an existing resource action. |
| Create resource action | Creates a resource action. |
| Delete resource action | Deletes a resource action. |

Table 3-11. XaaS Resource Actions (Continued)

| Workflow | Description |
|---------------------------|--------------------------------|
| Publish resource action | Publishes a resource action. |
| Unpublish resource action | Unpublishes a resource action. |

You can use the workflows in the **XaaS Resource Mappings** subdirectory to create and manage XaaS mappings to non-XaaS resources.

Table 3-12. XaaS Resource Mappings

| Workflow | Description |
|---------------------------|---------------------------------------------------------------------------------|
| Create a resource mapping | Map a catalog resource type to a vRealize Orchestrator type. |
| Delete a resource mapping | Delete a resource mapping. |
| Set a target criteria | Specify the conditions that determine the availability of the resource mapping. |

You can use the workflows in the **XaaS Server Configuration** subdirectory to manage the target Orchestrator instance.

Table 3-13. XaaS Server Configuration

| Workflow | Description |
|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Update Orchestrator server configuration | Modify the server settings, including port, host, user name, and password. |
| Validate Orchestrator server configuration | Verifies that the vRealize Orchestrator settings are valid. The workflow returns a value of TRUE if the configuration is valid, and FALSE if the configuration is not valid. |

You can use the workflows in the **XaaS Service Blueprints** subdirectory to create and manage XaaS blueprints.

Table 3-14. XaaS Blueprints

| Workflow | Description |
|-------------------------------|----------------------------------------|
| Clone a service blueprint | Creates a copy of a service blueprint. |
| Create a service blueprint | Creates a service blueprint. |
| Delete a service blueprint | Deletes a service blueprint. |
| Publish a service blueprint | Publishes a service blueprint. |
| Unpublish a service blueprint | Unpublishes a service blueprint. |

Using the vRealize Automation Plug-In Infrastructure Administration Workflows

You can use the infrastructure administration workflows to run basic operations. You use the extensibility package to customize vRealize Automation with the ability to call vRealize Orchestrator workflows either as part of the provisioning process, or by using custom operation menus.

You can find the infrastructure administration workflows on the **Workflows** view of the Orchestrator client, in the **Infrastructure Administration** subdirectory of the plug-in library.

You can use the infrastructure administration workflows to provision virtual machines and run basic create, read, update, or delete operations.

Table 3-15. Infrastructure Administration

| Workflow Name | Description |
|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Await virtual machine state change | <p>Awaits a state change for a set of virtual machines. If all virtual machines are in the success state, a trigger is called and the workflow ends successfully. If any of the specified virtual machines gets into the fail state, or does not exist, the workflow fails. You must enter the success and fail states selecting from the following options:</p> <ul style="list-style-type: none"> ■ Requested ■ AwaitingApproval ■ RegisterMachine ■ BuildingMachine ■ AddingDisks ■ MachineProvisioned ■ MachineActivated ■ InstallTools (VMware only) ■ On ■ Off ■ TurningOn ■ TurningOff ■ ShuttingDown ■ Suspending ■ Resetting ■ Rebooting ■ Expired ■ DeactivateMachine ■ UnprovisionMachine ■ Disposing ■ Finalized |
| Create an IaaS model entity | Creates and persists an entity for a specified vRealize Automation model. |
| Delete an IaaS model entity | Deletes a specified vRealize Automation model entity. |
| Invoke a post-provisioning action (deprecated) | Use the Request a resource action workflow. |
| Provision a virtual machine from a blueprint (removed in vRealize Automation 7.0) | Replaced by Request a catalog item or Request a catalog item with provisioning request. |
| Read an IaaS entity by custom filter | Reads a list of vRealize Automation entities by using a custom filter. If you do not specify a filter, all entities are returned as a result. |
| Read an IaaS entity by system query | Reads a list of vRealize Automation entities by using OData system filters. The system filters apply to the OData URI convention. |
| Read an IaaS model entity | Reads a vRealize Automation model entity by its ID. |
| Update an IaaS model entity | Updates a vRealize Automation model entity by its ID. |

You use the workflows in the **Extensibility** subdirectory to customize vRealize Automation with the ability to call vRealize Orchestrator workflows either as part of the provisioning process, or by custom operation menus.

Table 3-16. Extensibility

| Workflow Name | Description |
|-----------------------------|-------------------------------------------|
| Install vCO customization | Installs an Orchestrator customization. |
| Uninstall vCO customization | Uninstalls an Orchestrator customization. |

Table 3-16. Extensibility (Continued)

| Workflow Name | Description |
|--------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Import an IaaS Virtual Machine (replaces Register an IaaS Virtual Machine) | Imports a vRealize Automation virtual machine, maps it to a blueprint as a deployment resource type, and adds it to the catalog. |
| Import vCenter Virtual Machine (replaces Register vCenter Virtual Machine) | Imports a vSphere virtual machine, maps it to a blueprint as a deployment resource type, and adds it to the catalog items list. |
| Unregister virtual machine (removed in vRealize Automation 7.0) | No replacement workflow is provided. |
| Assign a menu operation to a blueprint and its virtual machines (deprecated) | Adds or updates a menu operation on virtual machines. Alternative non-deprecated workflows include Assign resource actions to an entitlement and Import a composite blueprint. |
| Assign a menu operation to virtual machines (deprecated) | Updates a vRealize Automation model entity by its ID. Alternative non-deprecated workflows include Assign resource actions to an entitlement and Import a composite blueprint. |
| Assign a state change workflow to a blueprint and its virtual machines | Adds or updates a state change workflow on several blueprints and their virtual machines. |
| Customize a menu operation (removed in vRealize Automation 7.0) | No replacement workflow is provided. |
| Remove a menu operation from a blueprint and its virtual machines (removed in vRealize Automation 7.0) | No replacement workflow is provided. |
| Remove a state change workflow from a blueprint and its virtual machines | Removes a state change workflow from a blueprint and its virtual machines. |

Create a vRealize Automation IaaS Model Entity

You can run a workflow to create a simple or complex vRealize Automation IaaS entity, such as a virtual machine reference to a user.

Procedure

- 1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
- 2 Click the **Workflows** view.
- 3 Select **Library > vRealize Automation > Infrastructure Administration**.
- 4 Right-click the **Create an IaaS model entity** workflow and select **Start workflow**.
- 5 Select a vRealize Automation host.
- 6 Enter the name of the model in the **Model name** text box.
- 7 Enter the name of the entity set, in the **Entity set name** text box.

You use scripting or a REST API to set the Simple properties, Links to complex properties, and HTTP headers properties.

- 8 Click **Submit** to run the workflow.

Read a vRealize Automation IaaS Model Entity

You can run a workflow to read a vRealize Automation IaaS model entity.

Procedure

- 1 From the drop-down menu in the Orchestrator client, select **Run** or **Design**.
- 2 Click the **Workflows** view.
- 3 Select **Library > vRealize Automation > Infrastructure Administration**.
- 4 Right-click **Read an IaaS model entity** and select **Start workflow**.
- 5 Select a vRealize Automation host.
- 6 Enter the name of the model in the **Model name** text box.
- 7 Enter the name of the entity set, in the **Entity set name** text box.
You use scripting or a REST API to set the HTTP headers property.
- 8 Click **Submit** to run the workflow.

Using the vRealize Automation Plug-In Requests Workflows

You can use the requests workflows to request catalog items and resource actions, and to complete or cancel work items.

You can access these workflows from the **Workflows** view of the vRealize Orchestrator client, in the **Requests** subdirectory of the plug-in library.

| Workflow | Description |
|-----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Cancel a work item | Cancels an active work item. You can use this workflow only if you are a system administrator. |
| Complete a work item | Finishes a work item. |
| Request a catalog item | Requests a catalog item for the user running the workflow. Use this workflow for XaaS blueprints. |
| Request a catalog item on behalf of a user | Sends a request for a catalog item on behalf of a user. You can use this workflow only for catalog items entitled to both you and the user on behalf of whom you are sending the request. Use this workflow for XaaS blueprints. |
| Request a catalog with provisioning request | Requests a composite blueprint as a catalog item for the user running the workflow. If you are providing customized input to the request, you must customize the workflow. Use this workflow for composite blueprints. |
| Request a resource action | Requests a resource action for a catalog item owned by the user running the workflow. |
| Request a resource action on behalf of a user | Sends a request for a resource action on behalf of a user. You can use this workflow only for resource actions entitled to both you and the user on behalf of whom you are sending the request. |
| Wait for a catalog item request | Waits for a catalog item request to finish. |
| Wait for a resource action request | Waits for a resource action request to finish. |
| Wait for a work item | Waits for a work item to finish. |

Using the vRealize Automation Plug-In Sample Workflows

You can use the sample workflows as examples, or as starting points for creating your own custom workflows.

You can find these workflows on the **Workflows** view of the vRealize Orchestrator client, in the **Sample** subdirectory of the plug-in library.

| Workflow Name | Description |
|-------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Create a permission | Provides a sample script that interacts with the authorization client and the permission service to create a permission in vRealize Automation. |
| Create a tenant | Creates a tenant with the same vRealize Automation host and Active Directory configuration as the default tenant. To run this workflow, select the vRealize Automation host that was added with your system administrator credentials. You can change the Active Directory settings before running the workflow. |
| List catalog items | Returns a list of catalog items for the selected tenant. |
| Print catalog item provisioning request as JSON | Retrieves the default request form for a catalog item and adds it to the console log. The log is formatted as JSON. The JSON file is a representation of the provisioning request for a catalog item. You can use the information to modify the Request a catalog item with a provisioning request workflow. |

Access the vRealize Automation Plug-In API

Orchestrator provides an API Explorer to allow you to search the vRealize Automation plug-in API and see the documentation for JavaScript objects that you can use in scripted elements.

For updated vRealize Automation API documentation, see <https://www.vmware.com/support/pubs/vcac-pubs.html>.

Procedure

- 1 Log in to the Orchestrator client as an administrator.
- 2 Select **Tools > API Explorer**.
- 3 Double-click the **vCAC** and **VCACCAFE** modules in the left pane to expand the hierarchical list of vRealize Automation plug-in API objects.

What to do next

You can copy code from API elements and paste it into scripting boxes. For more information about API scripting, see *Developing with VMware vRealize Orchestrator*.

For additional information about development best practices, see [vRealize Orchestrator Documentation](#).

Example vRealize Automation Plug-In Scripts

4

You can cut, paste, and edit the JavaScript examples provided to develop your own custom scripts for automating vRealize Automation tasks.

This chapter includes the following topics:

- [“CRUD Infrastructure Administration Tasks Example Scripts,”](#) on page 27
- [“Finding vRealize Automation Entities Example Scripts,”](#) on page 31
- [“Get a Resource Provisioned by vRealize Automation Example Script,”](#) on page 32
- [“Common Tasks Example Scripts,”](#) on page 33

CRUD Infrastructure Administration Tasks Example Scripts

You can cut, paste, and edit the JavaScript examples to write scripts for CRUD vRealize Automation tasks.

For more information about scripting in vRealize Orchestrator, see *Developing with VMware vRealize Orchestrator*.

Example: Create a vRealize Automation Model Entity

This example script performs the following actions:

- 1 Defines the model name and the entity set name.
- 2 Defines the properties of the host prefix.
- 3 Saves the host prefix entity.
- 4 Defines the properties of the provisioning group.
- 5 Defines the provisioning group as a link.
- 6 Saves the provisioning group entity, by linking it with the host name prefix.

Table 4-1. Input Variables

| Variable | Type |
|----------|---------------|
| host | vCAC:VcacHost |

```
var modelName = 'ManagementModelEntities.svc';
var entitySetName = 'HostNamePrefixes';
var links = null;
var headers = null;
//Create properties for prefix entity
var prefixInputProperties = {
```

```

    MachinePrefix:'test-prefix',
    NextMachineNo:1,
    MachineNumberLength:3
};
//Save the prefix
var prefixEntity = vCACEntityManager
    .createModelEntity(host.id, modelName, entitySetName, prefixInputProperties, links, headers);
entitySetName = 'ProvisioningGroups';
//Create properties for the provisioning group entity
inputProperties = {
    GroupName:'TestGroupName',
    GroupDescription:'This group was generated with a vCO workflow',
    AdministratorEmail:'test@test.com',
    AdContainer:'AD',
    IsTestGroup:false,
    Flags:2,
    GroupType:1};
//Add a reference to the newly created prefix entity
links = {
    HostNamePrefix:prefixEntity
};
//Save the provisioning group
var entity = vCACEntityManager.createModelEntity(host.id, modelName, entitySetName,
inputProperties, links, headers);

```

Example: Update a vRealize Automation Model Entity

This example script performs the following actions:

- 1 Gets the host ID from the provided entity.
- 2 Gets the model name from the provided entity.
- 3 Gets the entity set name from the provided entity.
- 4 Gets the entity ID from the provided entity.
- 5 Defines a set of properties that will be updated.
- 6 Starts the action responsible for updating the entity.

Table 4-2. Input Variables

| Variable | Type |
|--------------------|-------------|
| entity | vCAC:Entity |
| updatedDescription | String |

```

var hostId = entity.hostId;
var modelName = entity.modelName;
var entitySetName = entity.entitySetName;
var entityIdString = entity.keyString;
var links = null;
var headers = null;
var updateProperties = new Properties();
updateProperties.put("UserNameDescription", updatedDescription);

```

```
//Update the user description
System.getModule("com.vmware.library.vcac")
    .updateVCACEntity(hostId, modelName, entitySetName, entityIdString, updateProperties, links,
headers);
```

Example: Read a vRealize Automation Model Entity

This example script performs the following actions:

- 1 Defines the model name and the entity set name.
- 2 Defines the blueprint ID with a property object.
- 3 Reads the entity.

Table 4-3. Input Variables

| Variable | Type |
|-------------|---------------|
| host | vCAC:VcacHost |
| blueprintID | String |

```
var modelName = 'ManagementModelEntities.svc';
var entitySetName = 'VirtualMachineTemplates';
var links = null;
var headers = null;
//Create properties for the prefix entity
var blueprintId = {
    VirtualMachineTemplateID:blueprintId,
};
//Read the blueprint
var entity = vCACEntityManager
    .readModelEntity(host.id, modelName, entitySetName, blueprintId, headers);
```

Example: Delete a vRealize Automation Model Entity

This example script performs the following actions:

- 1 Gets the host ID from the provided entity.
- 2 Gets the model name from the provided entity.
- 3 Gets the entity set name from the provided entity.
- 4 Gets the entity ID from the provided entity.
- 5 Starts the action responsible for deleting the entity.

Table 4-4. Input Variables

| Variable | Type |
|----------|-------------|
| entity | vCAC:Entity |

```
var hostId = entity.hostId;
var modelName = entity.modelName;
var entitySetName = entity.entitySetName;
var entityKeyString = entity.keyString;
var headers = null;
//Delete the entity
System.getModule("com.vmware.library.vcac")
    .deleteVCACEntity(hostId, modelName, entitySetName, entityKeyString, headers);
```

Example: Read a vRealize Automation Entity by Custom Filter

This example script performs the following actions:

- 1 Defines the model name and the entity set name.
- 2 Defines the properties by which the entities are filtered.
- 3 Reads a list of entities.

Table 4-5. Input Variables

| Variable | Type |
|--------------|---------------|
| host | vCAC:VcacHost |
| templateName | String |

```
var modelName = 'ManagementModelEntities.svc';
var entitySetName = 'VirtualMachineTemplates';
var headers = null;
//Create properties for prefix entity
var properties = {
    VirtualMachineTemplateName:templateName,
};
//Read a list of entities
var entities = vCACEntityManager
    .readModelEntitiesByCustomFilter(host.id, modelName, entitySetName, properties, headers);
```

Example: Read a vRealize Automation Entity by System Query

This example script performs the following actions:

- 1 Defines the model name and the entity set name.
- 2 Defines the system queries by which the entities are filtered and selects the top ten results of all virtual machines, filtered by the machine state and component flag.
- 3 Reads a list of entities.

Table 4-6. Input Variables

| Variable | Type |
|----------|---------------|
| host | vCAC:VcacHost |

```
var modelName = 'ManagementModelEntities.svc';
var entitySetName = 'VirtualMachines';
var filter = "VirtualMachineState eq 'Off' and IsComponent eq true";
var orderBy = 'VirtualMachineName asc';
var top = 10; {
var skip = 0;
var headers = null;
var select = null;
var entities = vCACEntityManager
    readModelEntitiesBySystemQuery(host.id, modelName, entitySetName, filter, orderBy, select,
top, skip, headers);
```

Finding vRealize Automation Entities Example Scripts

You can cut, paste, and edit the JavaScript examples to write scripts for finding vRealize Automation entities by using the `vCACCAFEEntitiesFinder` scripting utility object.

For more information about scripting in vRealize Orchestrator, see *Developing with VMware vRealize Orchestrator*.

Example: Find Catalog Resources Filtered by Name

Table 4-7. Input Variables

| Variable | Type |
|----------|-------------------|
| host | vCACCAFE:VcacHost |

You can use one the following examples:

- This example script gets all catalog resources for the target host matching the query of *name_of_the_resource* by name and description.


```
var items = vCACCAFEEntitiesFinder.findCatalogResources(host, "name_of_the_resource");
```
- This example script performs the following actions:
 - a Gets the Consumer Resource service and invokes the get method passing as a Pageable parameter an instance of the `vCACCAFEPageOdataRequest` object.
 - b Creates the `vCACCAFEPageOdataRequest` object by providing an OData query as a single filter of the name attribute matching the *name_of_the_resource* string.

```
var service = host.createCatalogClient().getCatalogConsumerResourceService();

var filter = new Array();
filter[0] = vCACCAFEFILTERParam.equal("name",
vCACCAFEFILTERParam.string("name_of_the_resource"));
var query = vCACCAFEOdataQuery.query().addFilter(filter);

var items = service.getResourcesList(new vCACCAFEPageOdataRequest(query));
```

Example: Find Catalog Resources Filtered by Owner

This example script performs the following actions:

- 1 Gets the Consumer Resource service and invokes the get method passing as a Pageable parameter an instance of the `vCACCAFEPageOdataRequest` object.
- 2 Creates the `vCACCAFEPageOdataRequest` object by providing an OData query as a single filter of the owner/ref attribute matching the *user@domain.com* string.

The owners/ref attribute is a composition based on the internal structure and fields of the catalog resources. The `vCACCAFECatalogResource` entity has the owners attribute, which is a collection of `vCACCAFECatalogPrincipal` entities. The `vCACCAFECatalogPrincipal` entity has the ref property, which is a string representation of the principal id of the user.

```
var filter = new Array();
filter[0] = vCACCAFEFILTERParam.substringOf("owners/ref",
vCACCAFEFILTERParam.string("user@domain.com"));
var query = vCACCAFEOdataQuery.query().addFilter(filter);

var items = service.getResourcesList(new vCACCAFEPageOdataRequest(query));
```

Example: Find Catalog Resources Filtered by Name and Owner

This example script combines the OData queries from the previous two examples into a single one condition by using the `vCACCAFEFilterParam.and(array of conditions)` logic operator.

```
var conditions = new Array();
conditions[0] = vCACCAFEFilterParam.equal("name",
vCACCAFEFilterParam.string("name_of_the_resource_here"));
conditions[1] = vCACCAFEFilterParam.substringOf("owners/ref",
vCACCAFEFilterParam.string("user@domain.com"));

var filter = new Array();
filter[0] = vCACCAFEFilterParam.and(conditions);
var query = vCACCAFEodataQuery.query().addFilter(filter);

var items = service.getResourcesList(new vCACCAFEPageOdataRequest(query));
```

You can define other conditions by using different logic operators such as `vCACCAFEFilterParam.group(array of parameters)`, `vCACCAFEFilterParam.not(parameter)`, `vCACCAFEFilterParam.startsWith(id, string)`, `vCACCAFEFilterParam.endsWith(id, string)`, `vCACCAFEFilterParam.greaterThan(id, number)`, `vCACCAFEFilterParam.lessThan(id, number)`, and so on.

Get a Resource Provisioned by vRealize Automation Example Script

You can cut, paste, and edit the JavaScript example to write scripts for retrieving the actual entities of vRealize Automation provisioned resources.

The `CatalogResource` type represents the provisioned resources in vRealize Automation. This type has an attribute of `ProviderBinding` type which represents the relation between the catalog resource and its provider with the following attributes:

- `bindingId` - represents the identifier of the entity which is unique for the provider
- `providerRef` - identifies the catalog provider which corresponds directly to a service registered in the vRealize Automation component registry

For more information about scripting in vRealize Orchestrator, see *Developing with VMware vRealize Orchestrator*.

Example: Get a Virtual Machine Provisioned as a vRealize Automation Catalog Resource

This example uses a vRealize Automation host and its IaaS host as input parameters and, for a provided resource id, returns the corresponding IaaS virtual machine. The scripting code takes only catalog resources of `Virtual Machine` type that are provisioned by the `iaas-service` provider.

Table 4-8. Input Variables

| Variable | Type |
|-----------------------|--------------------------------|
| <code>vcacHost</code> | <code>vCACCAFE:VCACHost</code> |
| <code>iaasHost</code> | <code>vCAC:VCACHost</code> |

```
// Id of the catalog resource (or vCACCAFECatalogResource_instance.getId())
var resourceId = "c222629c-6f90-4458-8c92-8ece0ba06173";

var resource = vCACCAFEEntitiesFinder.getCatalogResource(vcacHost, resourceId);

var resourceType = resource.getResourceTypeRef().getLabel();
```



```

System.log("resource type: " + resourceType);

var providerBinding = resource.getProviderBinding();

var bindingId = providerBinding.getBindingId();
System.log("provider binding id: " + bindingId);

var provider = providerBinding.getProviderRef();
System.log("provider id: " + provider.getId());
System.log("provider name: " + provider.getLabel());

if ((resourceType == "Virtual Machine") && (provider.getLabel() == "iaas-service")) {
    System.log("It is an IaaS VM!");

    // IaaS virtual machine
    var vm = Server.findForType("vCAC:VirtualMachine", bindingId);
    System.log("IaaS VM id: " + vm.virtualMachineID);
    System.log("IaaS VM name: " + vm.displayName);

    // IaaS Entity
    var entity =
System.getModule("com.vmware.library.vcac").getVirtualMachineEntityFromId(iaasHost, bindingId);
    System.log("IaaS entity id: " + entity.keyString);
}

```

Common Tasks Example Scripts

You can cut, paste, and edit the JavaScript examples, or use them as samples to help you learn to develop your own scripts for common vRealize Automation tasks.

For more information about scripting in vRealize Orchestrator, see *Developing with VMware vRealize Orchestrator*.

Example: Create a vRealize Automation Advanced Service Blueprint

This example script performs the following actions:

- 1 Sets the vRealize Orchestrator workflow used to build the service blueprint.
- 2 Generates the content for the service blueprint based on the workflow.
- 3 Creates the service blueprint entity.
- 4 Publishes the service blueprint.

Table 4-9. Input Variables

| Variable | Type |
|----------|-------------------|
| host | vCACCAFE:VCACHost |

```

//ID of the workflow used to create the service blueprint
var workflowId = "44e42047-2fa0-4e4a-ba0c-12086540b28b";

var name = "MyBlueprint"
var description = "Blueprint description";
var workflowClient = host.createAdvancedDesignerClient().getAdvancedDesignerWorkflowService();

//Generate a service blueprint based on the workflow ID

```

```

var blueprint = workflowClient.generateServiceBlueprintByWorkflowId(workflowId);
blueprint.setTenant(host.tenant);
blueprint.setName(name);
blueprint.setDescription(description);

//Create the service blueprint
var blueprintService =
host.createAdvancedDesignerClient().getAdvancedDesignerServiceBlueprintService();
var uri = blueprintService.createServiceBlueprint(host.tenant , blueprint);

//Publish the service blueprint
var createdBlueprint = blueprintService.getServiceBlueprintByUri(uri);
blueprintService.updateServiceBlueprintStatus(host.tenant, createdBlueprint.getId(),
vCACCAFEDesignerPublishStatus.PUBLISHED);

```

Example: Create a vRealize Automation Approval Policy

This example script performs the following actions:

- 1 Gets the approval policy type.
- 2 Sets the user and group whose approval is required.
- 3 Sets the approval levels.
- 4 Defines the pre-provisioning approval phase.
- 5 Defines the post-provisioning approval phase.
- 6 Defines the approval policy specifications such as name, description, and type.
- 7 Creates the approval policy.
- 8 Publishes the approval policy. Once an approval policy is published, it becomes read-only.

Table 4-10. Input Variables

| Variable | Type |
|----------|-------------------|
| host | vCACCAFE:VCACHost |

```

// Get the type of approval policy by ID
var typeService = host.createApprovalClient().getApprovalApprovalPolicyTypeService();
var type = typeService.getApprovalPolicyType("com.vmware.cafe.catalog.request");

// Set the user and group required to complete the approval
var user = new vCACCAFEApprovalPrincipal();
user.setValue("user@domain.com");
user.setType(vCACCAFEApprovalPrincipalType.USER);

var group = new vCACCAFEApprovalPrincipal();
group.setValue("group@domain.com");
group.setType(vCACCAFEApprovalPrincipalType.GROUP);

// Set the level of the approval
var level = new vCACCAFEApprovalLevel();
level.setName("IT Approval Level");
level.setDescription("IT Approval Level description");
level.setApprovalMode(vCACCAFEApprovalMode.ALL);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(level, "getApprovers",

```

```

user);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(level, "getApprovers",
group);
level.setLevelNumber(1);

// Set pre-provisioning phase type and the phase of the approval
var phase1Type = new vCACCAFEApprovalPhaseType();
phase1Type.setId("com.vmware.cafe.catalog.request.pre");
phase1Type.setName("Pre-Provisioning type");
phase1Type.setDescription("Pre-Provisioning type description");
phase1Type.setPhaseOrder(1);

var phase1 = new vCACCAFEPhase();
phase1.setName("Pre-Provisioning");
phase1.setDescription("Pre provisioning phase");
phase1.setPhasetype(phase1Type);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(phase1, "getLevels",
level);

// Set post-provisioning phase type and the phase of the approval
var phase2Type = new vCACCAFEApprovalPhaseType();
phase2Type.setId("com.vmware.cafe.catalog.request.post");
phase2Type.setName("Post-Provisioning type");
phase2Type.setDescription("Post-Provisioning type description");
phase2Type.setPhaseOrder(1);

var phase2 = new vCACCAFEPhase();
phase2.setName("Post-Provisioning");
phase2.setDescription("Post provisioning phase");
phase2.setPhasetype(phase2Type);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(phase2, "getLevels",
level);

// Create the approval policy specifications
var spec = new vCACCAFEApprovalPolicy();
spec.setName("New Policy");
spec.setDescription("New Policy description");
spec.setPolicyType(type);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(spec, "getPhases", phase1);
System.getModule("com.vmware.library.vcaccafe.util").addElementToList(spec, "getPhases", phase2);

// Create the approval policy
var approvalPolicyService = host.createApprovalClient().getApprovalApprovalPolicyService();
var approvalPolicy = approvalPolicyService.createPolicy(spec);

// Publish the approval policy
approvalPolicy.setState(vCACCAFEApprovalPolicyState.PUBLISHED);
approvalPolicy = approvalPolicyService.update(approvalPolicy);
System.log("New approval policy id: " + approvalPolicy.getId());

```


Index

A

adding a vRealize Automation model entity **24**
API access **26**
audience **5**

C

CRUD operations, vRealize Automation **15, 16**

E

example **31, 32**

F

finding catalog resources **31**
functional prerequisites **10**

G

getting a provisioned virtual machine **32**

H

host
 configuring **11**
 managing **11**

I

laaS host, configuring **13**
installing
 external vRealize Orchestrator server **10**
 vRealize Automation plug-in **10**
Inventory **17**

R

reading a vRealize Automation model entity **25**

S

scriptable task elements **27, 33**
scripting **27, 33**

U

updated information **7**
using the vCACCAFEEntitiesFinder object **31**

V

vRealize Automation, CRUD operations **15, 16**
vRealize Automation host, configuring **12**
vRealize Automation plug-in
 configuring **11**
 installing **10**

installing in external vRealize Orchestrator
 server **10**

introduction **9**

vRealize Automation model entity
 adding **24**
 reading **25**

vRealize Orchestrator **9**

W

workflow library **15, 27**

workflows

 configuring **11**

 CRUD **17, 22, 26**

 extensibility workflows **22**

 laaS **22**

 model entity **17, 22, 26**

 requests workflows **25**

 standard workflows **17, 22, 26**

