Using the vRealize Orchestrator Plug-In for vRealize Automation 7.2

vRealize Orchestrator 7.2
vRealize Automation 7.2
vRealize Automation 7.2
vRealize Orchestrator 7.2

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-002397-00

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

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

Contents

Using the vRealize Orchestrator Plug-In for vRealize Automation

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

Intended Audience

The information in *Using the vRealize Orchestrator Plug-In for vRealize Automation* 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.

VMware vRealize Orchestrator Plug-In for vRealize Automation 简介 1

VMware vRealize Orchestrator Plug-in for vRealize Automation 支持在 vRealize Orchestrator 和 vRealize Automation 之间进行交互。

您可以使用 vRealize Automation 插件为以下 vRealize Automation 功能创建并运行工作流：

- XaaS 自定义资源和蓝图管理
- 目录项和资源的管理与申请
- 授权配置
- 批准策略配置
- 工作项交互
- vSphere 和 vCloud Director 虚拟机置备和置备后操作
- 在 vRealize Automation IaaS 模型上创建、读取、更新和删除 (CRUD) 操作

包含 vRealize Automation 插件的 vRealize Orchestrator 的角色

您可以使用 Orchestrator 客户端运行并创建工作流以及访问插件 API。您可以使用 vRealize Automation 系统中的嵌入式 vRealize Orchestrator 实例，或外部 vRealize Orchestrator 服务器。

vRealize Orchestrator 为 vRealize Automation 插件提供支持。vRealize Orchestrator 是一个开发与自动化处理平台，提供可扩展的工作流库，用于管理 VMware 云堆栈和第三方技术。

vRealize Orchestrator 可通过其开放插件架构与管理解决方案进行集成。

配置 vRealize Automation 插件 2

您需要添加 vRealize Automation 主机和 IaaS 主机来配置插件。

配置工作流

您可以使用**配置**工作流**类别**中的工作流来管理 vRealize Automation 主机。

vRealize Automation 主机

可以从 Orchestrator 客**户**端的**工作流视图访问**以下工作流，位于插件**库**的**配置**子目**录**中。

工作流名称	描述
添加 vRA 主机	将 vRealize Automation 主机添加到插件清**单**。**对**于租**户**管理和管理任**务**，您可以使用**清单视图**在每个租**户**上运行工作流。若要使租**户**使用**该**插件的完整功能，**请为**每个租**户创**建**专**用的 vRealize Automation 主机。
使用**组**件注册表添加 vRA 主机	将 vRealize Automation 主机添加到使用**"单**用**户**会**话"连**接的插件清**单**。您必**须**使用 vRealize Automation 系**统**管理**员**的凭据登**录**到 Orchestrator 客**户**端。 若要将此功能与外部 vRealize Orchestrator 服**务**器**结**合使用，必**须**在 vRealize Automation **组**件注册表中注册 Orchestrator 服**务**器。
添加 vRA 主机的 IaaS 主机	将**选**定 vRealize Automation 主机的 IaaS 主机添加到插件清**单**。
移除 vRA 主机	移除插件清**单**中的 vRealize Automation 主机。
更新 vRA 主机	更新插件清**单**中的 vRealize Automation 主机。
验证 vRA 主机	**验证** vRealize Automation 主机及其**连**接。

注意 如果 vRealize Orchestrator 服**务**器已在 vRealize Automation **组**件注册表中注册，**则**会自**动**添加名称**为**"默**认**"的 vRealize Automation 主机。"默**认**"主机使用"**单**用**户**会**话"连**接到默**认**租**户**。默**认**情况下，vRealize Automation 系**统**中的嵌入式 Orchestrator 服**务**器会在 vRealize Automation **组**件注册表中注册。

vRealize Automation IaaS 主机

您可以从 Orchestrator 客**户**端的**工作流视图访问**以下工作流，位于插件**库**的**基础架构管理 > 配置**子目**录**中。

默**认**情况下，vRealize Automation 系**统**中的嵌入式 vRealize Orchestrator 服**务**器会在 vRealize Automation **组**件注册表中注册。

工作流名称	描述
添加 IaaS 主机	将 vRealize Automation IaaS 主机添加到插件清**单**。此工作流的功能与"添加 vRA 主机的 IaaS 主机"相同，但不需要 vRealize Automation 主机。
移除 IaaS 主机	移除插件清**单**中的 vRealize Automation IaaS 主机。

工作流名称	描述
更新 IaaS 主机	更新插件清单中的 vRealize Automation IaaS 主机。
验证 IaaS 主机	验证 vRealize Automation IaaS 主机及其连接。

添加 vRealize Automation 主机

可以运行工作流以添加 vRealize Automation 主机并配置主机连接参数。

步骤

1 从 Orchestrator 客户端的下拉菜单中，选择运行或设计。

2 单击工作流视图。

3 展开库 > vRealize Automation > 配置。

4 右键单击添加 vRA 主机工作流，然后选择启动工作流。

5 在主机名称文本框中输入主机的唯一名称。

6 在主机 URL 文本框中输入主机的 URL 地址。

例如：*https://hostname*。

7 在租户文本框中输入租户的名称。

要使租户使用该插件的完整功能，请为每个租户创建专用的 vRealize Automation 主机。

8 选择是否自动安装 SSL 证书而无需用户确认。

9 （可选） 要配置 vRealize Orchestrator 等待来自 vRealize Automation 的连接或响应所用的时间长度，请在连接超时 (秒) 和操作超时 (秒) 文本框中输入超时间隔。

10 从会话模式下拉菜单中选择与主机的连接类型。

选项	操作
共享会话	在身份验证用户名文本框和身份验证密码文本框中输入 vRealize Automation 用户的凭据。
每用户会话	使用当前已登录的用户的凭据进行连接。您必须使用 vRealize Automation 系统管理员的凭据登录到 Orchestrator 客户端。 要将此选项与外部 vRealize Orchestrator 服务器结合使用，您必须在 vRealize Automation 组件注册表中注册 Orchestrator 服务器。

11 单击提交。

下一步

添加 vRealize Automation 基础架构管理主机。

添加 IaaS 主机

可以运行工作流以添加 vRealize Automation 主机的 IaaS 主机并配置连接参数。

步骤

1 从 Orchestrator 客户端的下拉菜单中，选择运行或设计。

2 单击工作流视图。

3 展开库 > vRealize Automation > 基础架构管理 > 配置。

4 右键单击添加 IaaS 主机，然后选择启动工作流。

5 从 vCAC 主机下拉菜单中选择要为其配置 IaaS 主机的 vRealize Automation 主机。

6 在**主机名称**文本框中**输**入主机的唯一名称。

7 **输**入安装了 Model Manager 的**计**算机的 URL。

例如：https://*model_manager_machine.com*。

8 要安装 SSL **证书**，**请选择是**。

9 要使用代理**访问** Model Manager **计**算机，**请选择是**。

如果**选择**此**选项**，**则必须**在下一**页**上提供代理主机和代理端口。

10 **单击下一步**。

11 如果要配置**显**式代理，**请**提供代理主机 URL 和端口。

12 **单击下一步**。

13 要配置您自己的超**时值**，**请单击否**。

14 （可**选**） 要配置 vRealize Orchestrator 等待来自 vRealize Automation 的**连**接或响**应**所用的**时间长**度，**请**在**连接超时 (秒)** 和**操作超时 (秒)** 文本框中**输**入超**时间**隔。

15 **单击下一步**。

16 **选择**主机的身份**验证类**型。

选项	**描述**
SSO	**选择**此**选项**可使用 vCenter Single Sign-On。
NTLM	**选择**此**选项**可**仅**在您的 Active Directory 基**础**架构依**赖**于 NT LAN 管理器 (NTLM) 身份**验证时**启用基于 NTLM **协议**的身份**验证**。 如果**选择**此**选项**，**则必须**指定其他 NTLM 凭据和身份**验证选项**。

17 如果已**选择** NTLM，**请单击下一步**，然后**输**入 Workstation **计**算机的名称和 NetBIOS 域名。

18 **单击提交**。

使用 vRealize Automation 插件工作流 3

vRealize Automation 插件工作流库中包含多种可用于常规任务（例如与目录交互、管理基础架构以及创建租户和服务）的工作流。

您可以使用自定义 HTTP 标头（例如 vRealize Automation 特定标头“任务”和“身份”），并将其应用到 CRUD、置备和置备后等工作流中。

本章讨论了以下主题：

移除操作限制

从版本 7.0 开始，某些创建、读取、更新和删除操作会受到限制。如果使用了先前版本工作流的操作，则这些操作将无法用于 7.0 和更高版本。您可以将工作流更新为支持的操作，或可以重新启动所需的操作。

若要重新启用这些操作，必须将要启用的操作从 `operations.properties` 文件移除。有关文件中的操作列表，请参见“受限制的操作”第 14 页，。

步骤

1 在 vRealize Orchestrator 的下拉菜单中，选择设计。

2 单击资源视图。

3 在资源层次结构中，展开**库 > VCAC > Util**。

4 创建备份并修改 `operations.properties` 文件。

   a 右键单击 `operations.properties`，然后选择保存为文件。

   b 将副本另存为备份。

   c 创建新副本并删除想要重新启用的操作。

   d 保存新文件。

5 替换 vRealize Orchestrator 中的现有文件。

   a 在 vRealize Orchestrator 中，右键单击 **Util** 文件夹并单击**导入资源**。

   b 浏览到新版本的 `operations.properties` 文件，然后单击**打开**。

   c 单击**替换一次**以保存修改后的版本。

6 重新启动 vRealize Orchestrator 服务器。

7 选择 `operations.properties` 文件，然后单击**查看器选项**卡。

8 验证要启用的操作是否已不在文件中。

从文件中移除的操作现在可用于较旧的工作流。

下一步

创建新工作流时，避免使用受限制的操作。

受限制的操作

`operations.properties` 文件的内容包含受限制的操作。若要重新启用该操作，必须将其从文件中移除。

以下文本为 `operations.properties` 文件的默认版本。若要重新启用某个操作，请参见“移除操作限制”第 13 页，。

```
#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
```

使用 vRealize Automation 插件清单

您可以使用**清单视图**在 vRealize Automation 对象上运行工作流。

要显示可供清单对象使用的工作流，请导航到**工具 > 用户首选项 > 清单**，然后选中**在清单中使用上下文菜单**复选框。启用此选项后，当您右键单击 Orchestrator 清单中的对象时，将显示可供该对象使用的所有工作流。

使用 vRealize Automation 插件管理工作流

您可以使用管理工作流来管理 vRealize Automation 服务、租户、批准策略、授权、业务组、目录项和高级服务组件。

部分工作流包含 vRealize Automation 主机的输入参数 `vCACCAFE:VCACHost`。配置 vRealize Automation 主机连接的方式决定了用户运行工作流时各种角色的应用方式。

- 如果将连接配置为共享会话，则共享会话的用户帐户必须具有运行工作流所需的角色。
- 如果将连接配置为单用户会话，运行工作流的每个用户必须具有所需的角色，与在 vRealize Automation 用户界面中一样。

您可以在 vRealize Orchestrator 客户端的**工作流视图**上找到以下工作流，位于**库 > vRealize Automation > 管理**子目录中。

您可以使用**批准策略**子目录中的工作流创建和管理批准策略。

表 3-1 批准策略

工作流	描述
激活批准策略	激活批准策略。批准策略在激活后会变为只读。
添加批准级别	向批准添加始终需要的批准级别。您必须为审批者选择特定用户和组。
复制批准策略	复制批准策略。
创建批准策略	创建不含级别或审批者的批准策略草稿。若要为自己的策略创建批准级别并指定审批者，请运行“添加批准级别”工作流。

表 3-1 批准策略 （续）

工作流	描述
取消激活批准策略	取消激活批准策略。您还可以删除与批准策略相关的所有现有授权。
删除批准策略	删除草稿状态的批准策略。活动的批准策略为只读。

您可以使用**业务组**子目录中的工作流创建和管理业务组以及业务组自定义属性。

表 3-2 业务组

工作流	描述
添加自定义属性	将自定义属性添加到业务组。
创建业务组	创建业务组。
删除业务组	删除业务组。
删除自定义属性	移除业务组中的自定义属性。
更新业务组	更新业务组的详细信息，例如默认计算机前缀、活动目录容器和用户角色。
更新自定义属性	更新业务组的自定义属性。

“管理”子目录包含**业务组（已弃用）**子目录，可与 vRealize Automation 7.0 之前的版本一起使用。请使用主文件夹中同名的工作流。

您可以使用**目录项**子目录中的工作流来管理目录项。

表 3-3 目录项

工作流	描述
激活目录项	激活目录项。您必须激活一个目录项并将其分配给服务，然后用户才能请求该服务。
向服务分配目录项。	向服务分配目录项。您必须激活一个目录项并将其分配给服务，然后用户才能请求该服务。
取消激活目录项	取消激活目录项并将其从服务目录中移除，从而使用户无法再请求。

您可以使用**复合蓝图**子目录中的工作流来管理设计画布中的复合蓝图。

表 3-4 复合蓝图

工作流	描述
删除复合蓝图	删除设计蓝图列表中未发布的蓝图。
导入复合蓝图	从 YAML 文件导入复合蓝图。
发布复合蓝图	发布处于草稿状态的复合蓝图。
取消发布复合蓝图	取消发布已发布的复合蓝图。

内容子目录工作流已弃用。使用 Cloud Client 执行导入和导出操作。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。

表 3-5 内容

工作流	描述
导出内容（已弃用）	使用 Cloud Client 执行导入和导出操作。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。
导入内容（已弃用）	使用 Cloud Client 执行导入和导出操作。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。
传输内容（已弃用）	使用 Cloud Client 执行导入和导出操作。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。
验证内容（已弃用）	使用 Cloud Client 执行导入和导出操作。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。

您可以使用授权子目录中的工作流创建和管理授权。

表 3-6 授权

工作流	描述
激活授权	激活授权。
向授权分配目录项	向授权分配一个或多个目录项。您还可以使用该工作流分配批准策略。
向授权分配即时操作	向授权分配一个或多个即时操作。即时操作不会创建请求。
向授权分配资源操作	向授权分配一个或多个资源操作。您还可以使用该工作流分配批准策略。
向授权分配服务	向授权分配一个或多个服务。您还可以使用该工作流分配批准策略。
向授权分配用户和组	向授权分配一个或多个用户或组。
创建授权（已弃用）	创建授权。对子租户使用“创建授权”。
为子租户创建授权	创建授权。
取消激活授权	取消激活授权。
取消分配给授权的用户和组	移除某个授权的用户列表中的用户和组。

您可以使用属性子目录中的工作流来管理属性定义和属性组。为避免与 vRealize Automation 属性发生冲突，所有自定义属性名称都应使用前缀，例如公司名称或功能名称，后面加个点 (.)。

表 3-7 属性定义

工作流	描述
创建属性定义	创建自定义属性。
删除属性定义	删除自定义属性。

属性组是属性定义的集合。

表 3-8 属性组

工作流	描述
将属性添加到组	将定义的自定义属性添加到组。
创建属性组	创建可在其中添加定义的自定义属性的属性组。
删除属性组	删除属性组。
移除组中的属性	移除属性组中定义的自定义属性。
更新属性组	修改属性组的名称或描述。
更新组中的属性	修改属性组中属性的名称、值和行为。

您可以使用**服务**子目录中的工作流来管理服务。

表 3-9 服务

工作流	描述
激活服务	激活服务。
向服务分配目录项	向服务分配一个或多个目录项。
复制服务	复制服务。
创建服务	创建服务。
取消激活服务	取消激活服务。
删除服务	删除服务。

您可以使用**租户**子目录中的工作流来创建和管理租户。

身份存储工作流已弃用。替换工作流能可自如应对 Directories Management API 的 vRealize Automation 的更改。

表 3-10 租户

工作流	描述
添加管理员	向租户添加一个或多个租户管理员和基础架构管理员。
将身份存储添加到租户	向 vRealize Automation 主机的租户添加身份存储。仅当作为系统管理员配置租户时，才能运行该工作流。
将身份存储添加到租户（已弃用）	使用“将身份存储添加到租户”工作流。
将身份存储添加到 vCAC 主机	向配置为 vRealize Automation 主机的租户添加身份存储。仅当作为系统管理员为租户配置身份存储时，才能运行该工作流。
将身份存储添加到 vCAC 主机（已弃用）	使用“将身份存储添加到 vCAC 主机”。
创建租户	创建租户。您必须选择使用系统管理员凭据添加的 vRealize Automation 主机。
删除租户的身份存储	删除 vRealize Automation 主机租户的身份存储。仅当作为系统管理员配置租户时，才能运行该工作流。
删除 vCAC 主机中的身份存储	删除配置为 vRealize Automation 主机的租户的身份存储。仅当作为系统管理员为租户配置身份存储时，才能运行该工作流。
删除租户	删除租户。
移除管理员	移除租户的一个或多个租户管理员和基础架构管理员。
更新租户的身份存储	更新 vRealize Automation 主机租户的现有身份存户。仅当作为系统管理员配置租户时，才能运行该工作流。
更新租户的身份存储（已弃用）	使用“更新租户的身份存储”工作流。

表 3-10 租户 （续）

工作流	描述
更新 vCAC 主机的身份存储	更新配置为 vRealize Automation 主机的租户的身份存储。仅当作为系统管理员为租户配置身份存储时，才能运行该工作流。
更新 vCAC 主机的身份存储（已弃用）	使用“更新 vCAC 主机的身份存储”工作流。
更新租户	更新现有租户的名称、描述和联系人电子邮件地址。

您可以使用**工作流订阅**子目录中的工作流来管理事件工作流订阅。

表 3-11 工作流订阅

工作流	描述
删除工作流订阅	删除未发布的工作流订阅。此工作流适用于系统和租户工作流订阅。
导出系统工作流订阅	导出系统工作流订阅并将其以 JSON 格式另存为 vRealize Orchestrator 资源元素。 系统工作流订阅是一个专用的工作流订阅，用于对系统事件和所有租户中的事件做出响应。
导出租户工作流订阅	导出租户工作流订阅并将其以 JSON 格式另存为资源元素。 一个专用工作流订阅用于运行租户特定的工作流。
导入系统工作流订阅	从 JSON 文件导入系统工作流订阅。系统工作流订阅会在遇到系统事件时触发，并且可以跨租户触发。
导入租户工作流订阅	从 JSON 文件导入已导出的工作流订阅。这些工作流订阅特定于租户。
发布工作流订阅	发布处于草稿或未发布状态的工作流订阅。此工作流适用于系统和租户工作流订阅。
注册系统工作流订阅	创建系统工作流订阅，其中包含超时值和优先级值。
注册租户工作流订阅	创建租户特定的工作流订阅，其中包含超时值和优先级值。
取消发布工作流订阅	取消发布已发布的工作流订阅。此工作流适用于系统和租户工作流订阅。
更新工作流订阅	更改名称、描述、vRealize Orchestrator 工作流、订阅条件、超时值、状态值和优先级值。您无法更新事件主题或阻止状态。

您可以使用 **XaaS 自定义资源**子目录中的工作流来创建和删除 XaaS 自定义资源。

表 3-12 XaaS 自定义资源

工作流	描述
创建自定义资源	创建自定义资源。
删除自定义资源	移除自定义资源。

您可以使用 **XaaS 资源操作**子目录中的工作流来创建和管理 XaaS 资源操作。

表 3-13 XaaS 资源操作

工作流	描述
克隆资源操作	创建现有资源操作的副本。
创建资源操作	创建资源操作。
删除资源操作	删除资源操作。

表 3-13 XaaS 资源操作 （续）

工作流	描述
发布资源操作	发布资源操作。
取消发布资源操作	取消发布资源操作。

您可以使用 **XaaS 资源映射**子目录中的工作流来创建和管理非 XaaS 资源的 XaaS 映射。

表 3-14 XaaS 资源映射

工作流	描述
创建资源映射	将目录资源类型映射到 vRealize Orchestrator 类型。
删除资源映射	删除资源映射。
设置目标标准	指定确定资源映射可用性的条件。

您可以使用 **XaaS 服务器配置**子目录中的工作流来管理目标 Orchestrator 实例。

表 3-15 XaaS 服务器配置

工作流	描述
更新 Orchestrator 服务器配置	修改服务器设置，包括端口、主机、用户名和密码。
验证 Orchestrator 服务器配置	验证 vRealize Orchestrator 设置是否有效。如果配置有效，则工作流会返回 TRUE 值，反之则返回 FALSE。

您可以使用 **XaaS 服务蓝图**子目录中的工作流来创建和管理 XaaS 蓝图。

表 3-16 XaaS 蓝图

工作流	描述
克隆服务蓝图	创建服务蓝图的副本。
创建服务蓝图	创建服务蓝图。
删除服务蓝图	删除服务蓝图。
发布服务蓝图	发布服务蓝图。
取消发布服务蓝图	取消发布服务蓝图。

使用 vRealize Automation 插件基础架构管理工作流

您可以使用基础架构管理工作流来运行基本操作。您需要使用可扩展软件包对 vRealize Automation 进行自定义，使其能够在置备过程中调用 vRealize Orchestrator 工作流，或使用自定义操作菜单进行调用。

可以从 Orchestrator 客户端的**工作流视图**中查找基础架构管理工作流，位于插件库的**基础架构管理**子目录中。

您可以使用基础架构管理工作流来置备虚拟机并运行基本的创建、读取、更新或删除操作。

表 3-17 基础架构管理

工作流名称	描述
等待虚拟机状态更改	等待一组虚拟机的状态发生更改。如果所有虚拟机都处于成功状态，则会调用触发器并且成功结束工作流。如果指定的虚拟机中有任何一台处于失败状态或不存在，则工作流失败。您必须选择以下选项输入成功和失败状态： ■ **Requested** ■ **AwaitingApproval** ■ **RegisterMachine** ■ **BuildingMachine** ■ **AddingDisks** ■ **MachineProvisioned** ■ **MachineActivated** ■ **InstallTools**（仅 VMware） ■ **On** ■ **Off** ■ **TurningOn** ■ **TurningOff** ■ **ShuttingDown** ■ **Suspending** ■ **Resetting** ■ **Rebooting** ■ **Expired** ■ **DeactivateMachine** ■ **UnprovisionMachine** ■ **Disposing** ■ **Finalized**
创建 IaaS 模型实体	为指定的 vRealize Automation 模型创建并保留实体。
删除 IaaS 模型实体	删除指定的 vRealize Automation 模型实体。
调用置备后操作（已弃用）	使用“请求资源操作”工作流。
从蓝图置备虚拟机（已在 vRealize Automation 7.0 中移除）	已替换为“请求目录项”或“通过置备请求来请求目录项”。
使用自定义筛选器读取 IaaS 实体	使用自定义筛选器读取 vRealize Automation 实体列表。如果未指定筛选器，则结果会返回所有实体。
使用系统查询读取 IaaS 实体	使用 OData 系统筛选器读取 vRealize Automation 实体列表。系统筛选器适用于 OData URI 约定。
读取 IaaS 模型实体	使用模型实体 ID 读取 vRealize Automation 模型实体。
更新 IaaS 模型实体	更新 vRealize Automation 模型实体（按 ID）。

您需要使用**可扩展性**子目录中的工作流对 vRealize Automation 进行自定义，使其能够在置备过程中调用 vRealize Orchestrator 工作流，或通过自定义操作菜单进行调用。

子目录中还包含可用于管理 IaaS 凭据、端点、企业组、计算机前缀以及其他实体的的工作流。

表 3-18 可扩展性

工作流名称	描述
安装 vCO 自定义	安装 Orchestrator 自定义，包含自定义的状态更改工作流和菜单操作工作流。
卸载 vCO 自定义	卸载 Orchestrator 自定义，包含自定义的状态更改工作流和菜单操作工作流。
更改 IaaS 虚拟机的预留	更改受管虚拟机的属性（例如预留和业务组）。

表 3-18 可扩展性 （续）

工作流名称	描述
导入 IaaS 虚拟机（已弃用）	使用 Cloud Client。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。
导入 vCenter 虚拟机（已弃用）	使用 Cloud Client。Cloud Client 的下载及相关文档可从 https://developercenter.vmware.com/tool/cloudclient 获取。
取消注册虚拟机（已在 vRealize Automation 7.0 中移除）	不会提供替换工作流。
向蓝图及其虚拟机分配菜单操作（已弃用）	在虚拟机上添加或更新菜单操作。 备用的未弃用工作流包括“向授权分配资源操作”以及“导入复合蓝图”。
向虚拟机分配菜单操作（已弃用）	更新 vRealize Automation 模型实体（按 ID）。 备用的未弃用工作流包括“向授权分配资源操作”以及“导入复合蓝图”。
向蓝图及其虚拟机分配状态更改工作流（已弃用）	已替换为 vRealize Automation 中的事件代理订阅。
自定义菜单操作（已在 vRealize Automation 7.0 中移除）	不会提供替换工作流。
从蓝图及其虚拟机中移除菜单操作（已在 vRealize Automation 7.0 中移除）	不会提供替换工作流。
从蓝图及其虚拟机中移除状态更改工作流	从蓝图及其虚拟机中移除状态更改工作流。

创建 vRealize Automation IaaS 模型实体

您可以运行工作流来创建简单或复杂 vRealize Automation IaaS 实体，例如用户的虚拟机参考。

步骤

1 从 Orchestrator 客户端的下拉菜单中，选择运行或设计。

2 单击工作流视图。

3 展开**库 > vRealize Automation > 基础架构管理**。

4 右键单击**创建 IaaS 模型实体**工作流，然后选择**启动工作流**。

5 选择一个 vRealize Automation 主机。

6 在**模型名称**文本框中输入模型的名称。

7 在**实体集名称**文本框中输入实体集的名称。

您需要使用脚本或 REST API 来设置简单属性、复杂属性的链接以及 HTTP 标头属性。

8 单击**提交**以运行工作流。

读取 vRealize AutomationIaaS 模型实体

您可以运行工作流来读取 vRealize Automation IaaS 模型实体。

步骤

1 从 Orchestrator 客户端的下拉菜单中，选择运行或设计。

2 单击工作流视图。

3 展开**库 > vRealize Automation > 基础架构管理**。

4 右键单击**读取 IaaS 模型实体**，然后选择**启动工作流**。

5 选择一个 vRealize Automation 主机。

6 在**模型名称**文本框中**输**入模型的名称。

7 在**实体集名称**文本框中**输**入**实**体集的名称。

您需要使用脚本或 REST API 来**设**置 HTTP **标头**属性。

8 **单击提交**以运行工作流。

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.

A work item requires user input or action. For example, a workflow interaction, approval action, or responding to a reclamation request.

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 based on provided user input.
Request a catalog item	Requests a catalog item for the user running the workflow. If you need a workflow to request a composite blueprint, use the Request a catalog with provisioning request workflow.
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 a on behalf of whom you are sending the request.
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 t 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 the user on behalf of whom you are sending the request.

Workflow	Description

Request a resource action with a request template	Requests a resource action that includes complex parameters. The best practice is to duplicate the workflow and then custom the action. You can use the workflow to pass complex parameters or hidden parameters that you do not want to appear on th form. One of the primary applications of this workflow is to customize the IaaS reconfigure virtual machine action.
	To create a reconfigure operation on a virtual machine, you must create a copy of the workflow and then modify the script. C the parameters that appear in vRealize Orchestrator and set the `Cafe.Shim.VirtualMachine.Reconfigure.Requestor` pa This parameter is used for logging and it must not be empty. See the following example.
	`var requestTemplate = vCACCAFERequestsHelper.getRequestForResourceAction(operation)`
	
	`var jsonData = vCACCAFERequestsHelper.getResourceActionRequestData(requestTemplate);`
	`var json = JSON.parse(jsonData);`
	`//Change cpu example`
	`json.cpu = 2;`
	
	`//This is a property needed for the Reconfigure IaaS operation:`
	`json["Cafe.Shim.VirtualMachine.Reconfigure.Requestor"] = 1;`
	`vCACCAFERequestsHelper.setResourceActionRequestData(requestTemplate, JSON.stringify(json));`
	
	`request =`
	`System.getModule("com.vmware.library.vcaccafe.request").requestResourceActionWithRequestTemplate(op`
	`requestTemplate);`

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.

使用 vRealize Automation 插件示例工作流

您可以将示例工作流用作示例，或用作创建自定义工作流的起点。

您可以在 vRealize Orchestrator 客户端的**工作流视图**上找到以下工作流，位于插件**库**的**示例**子目录中。

工作流名称	描述
创建权限	提供可与授权客户端交互的示例脚本以及可在 vRealize Automation 中创建权限的权限服务。
创建租户	创建与默认租户具有相同 vRealize Automation 主机和 Active Directory 配置的租户。若要运行此工作流，请选择使用您的系统管理员凭据添加的 vRealize Automation 主机。您可以先更改 Active Directory 设置，然后再运行工作流。
列出目录项	返回选定租户的目录项列表。
打印目录项置备请求（JSON 格式）	检索目录项的默认请求表单，并将其以 JSON 格式添加到控制台日志中。您可以使用这些数据来自定义置备请求。您可以使用该信息来修改通过置备请求来请求目录项工作流。

访问 vRealize Automation 插件 API

Orchestrator 提供 API Explorer，您可用其搜索 vRealize Automation 插件 API 并查看相关文档，了解可用于脚本元素的 JavaScript 对象。

有关更新的 vRealize Automation API 文档，请参见 https://www.vmware.com/support/pubs/vcac-pubs.html。

步骤

1 以管理员身份登录到 Orchestrator 客户端。

2 **选择工具 > API Explorer**。

3 双**击**左**侧**窗格中的 **vCAC** 和 **VCACCFE** 模**块**以展开 vRealize Automation 插件 API **对**象的**层**次**结**构列表。

下一步

可以从 API 元素复制代**码**并将代**码**粘**贴**到脚本**编**写框中。有关 API 脚本**编**写的更多信息，**请**参**见** *《使用 VMware vRealize Orchestrator **进**行开**发**》*。

有关开**发**最佳做法的其他信息，**请**参**见** vRealize Orchestrator 文档。

4 示例 vRealize Automation 插件脚本

您可以剪切、粘贴和编辑所提供的 JavaScript 示例以开发自己的自定义脚本，用于自动处理 vRealize Automation 任务。

本章讨论了以下主题：

CRUD 基础架构管理任务示例脚本

您可以剪切、粘贴和编辑 JavaScript 示例来编写脚本，用于执行 CRUD vRealize Automation 任务。

有关 vRealize Orchestrator 中脚本编写的更多信息，请参见 *《使用 VMware vRealize Orchestrator 进行开发》*。

示例：创建 vRealize Automation 模型实体

此示例脚本会执行以下操作：

1 定义模型名称和实体集名称。

2 定义主机前缀的属性。

3 保存主机前缀实体。

4 定义置备组的属性。

5 将置备组定义为链接。

6 将置备组实体与主机名前缀链接，保存置备组实体。

表 4-1 输入变量

变量	类型
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);
```

示例：更新 vRealize Automation 模型实体

此示例脚本会执行以下操作：

1 获取所提供实体的主机 ID。

2 获取所提供实体的模型名称。

3 获取所提供实体的实体集名称。

4 获取所提供实体的实体 ID。

5 定义将要更新的一组属性。

6 启动用于更新实体的操作。

表 4-2 输入变量

变量	类型
entity	vCAC:Entity
updatedDescription	字符串

```
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);
```

示例：读取 vRealize Automation 模型实体

此示例脚本会执行以下操作：

1 定义模型名称和实体集名称。

2 定义带有属性对象的蓝图 ID。

3 读取实体。

表 4-3 输入变量

变量	类型
host	vCAC:VcacHost
blueprintID	字符串

```
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);
```

示例：删除 vRealize Automation 模型实体

此示例脚本会执行以下操作：

1 获取所提供实体的主机 ID。

2 获取所提供实体的模型名称。

3 获取所提供实体的实体集名称。

4 获取所提供实体的实体 ID。

5 启动用于删除实体的操作。

表 4-4 输入变量

变量	类型
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);
```

示例：使用自定义筛选器读取 vRealize Automation 实体

此示例脚本会执行以下操作：

1 定义模型名称和实体集名称。

2 定义用来筛选实体的属性。

3 读取实体列表。

表 4-5 输入变量

变量	类型
host	vCAC:VcacHost
templateName	字符串

```
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);
```

示例：使用系统查询读取 vRealize Automation 实体

此示例脚本会执行以下操作：

1 定义模型名称和实体集名称。

2 定义用来筛选实体的系统查询，并选择所有虚拟机的前十个结果（按虚拟机状态和组件标记筛选）。

3 读取实体列表。

表 4-6 输入变量

变量	类型
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);
```

查找 vRealize Automation 实体示例脚本

您可以剪切、粘贴和编辑 JavaScript 示例以编写脚本，用于通过 vCACCAFEEntitiesFinder 脚本实用程序对象来查找 vRealize Automation 实体。

有关 vRealize Orchestrator 中脚本编写的更多信息，请参见 *《使用 VMware vRealize Orchestrator 进行开发》*。

示例：查找按名称筛选的目录资源

表 4-7 输入变量

变量	类型
host	vCACCAFE:VcacHost

您可以使用以下任一示例：

- 此示例脚本会获取目标主机上与 *name_of_the_resource* 查询匹配（按名称和描述）的所有目录资源。

```
var items = vCACCAFEEntitiesFinder.findCatalogResources(host, "name_of_the_resource");
```

- 此示例脚本会执行以下操作：

  a 获取“用户资源”服务并调用 get 方法，将 Pageable 对象的实例作为 vCACCAFEPageOdataRequest 参数进行传递。

  b 创建 vCACCAFEPageOdataRequest 对象时使用 OData 查询作为单个筛选器用于筛选与 *name_of_the_resource* 字符串匹配的 name 属性。

```
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));
```

示例：查找按所有者筛选的目录资源

此示例脚本会执行以下操作：

1 获取“用户资源”服务并调用 get 方法，将 vCACCAFEPageOdataRequest 对象的实例作为 Pageable 参数进行传递。

2 创建 vCACCAFEPageOdataRequest 对象时使用 OData 查询作为单个筛选器用于筛选与 *user@domain.com* 字符串匹配的 owner/ref 属性。

   owners/ref 属性是基于内部结构和目录资源字段的组合。vCACCAFECatalogResource 实体具有 owners 属性，即 vCACCAFECatalogPrincipal 实体的集合。vCACCAFECatalogPrincipal 实体具有 ref 属性，即用户主体 ID 的字符串表现形式。

```
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));
```

示例：查找按名称和所有者筛选的目录资源

此示例脚本使用 vCACCAFEFilterParam.and(array of conditions) 逻辑运算符将之前两个示例的 OData 查询合并为一个条件。

```
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));
```

您可以使用其他逻辑运算符（例如 vCACCAFEFilterParam.group(array of parameters)、vCACCAFEFilterParam.not(parameter)、vCACCAFEFilterParam.startsWith(id, string)、vCACCAFEFilterParam.endsWith(id, string)、vCACCAFEFilterParam.greaterThan(id, number)、vCACCAFEFilterParam.lessThan(id, number) 等）定义其他条件。

获取 vRealize Automation 示例脚本置备的资源

您可以剪切、粘贴和编辑 JavaScript 示例来编写脚本，用于检索 vRealize Automation 置备的资源的实际实体。

CatalogResource 类型表示 vRealize Automation 中已置备的资源。此类型具有 ProviderBinding 类型的属性，代表具有以下属性的目录资源及其提供程序之间的关系：

- bindingId - 表示提供程序独具的实体的标识符
- providerRef - 用于识别与 vRealize Automation 组件注册表中所注册的服务直接对应的目录提供程序

有关 vRealize Orchestrator 中脚本编写的更多信息，请参见 *《使用 VMware vRealize Orchestrator 进行开发》*。

示例：获得置备为 vRealize Automation 目录资源的虚拟机

此示例将 vRealize Automation 主机及其 IaaS 主机用作输入参数，并且针对提供的资源 ID，返回相应的 IaaS 虚拟机。脚本代码仅采用 Virtual Machine 类型的目录资源（由 iaas-service 提供程序置备）。

表 4-8 输入变量

变量	类型
vcacHost	vCACCAFE:VCACHost
iaasHost	vCAC:VCACHost

```
// 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);
}
```

常规任务示例脚本

您可以剪切、粘贴和编辑 JavaScript 示例，或作为示例来帮助开发常规 vRealize Automation 任务的脚本。

有关 vRealize Orchestrator 中脚本编写的更多信息，请参见 *《使用 VMware vRealize Orchestrator 进行开发》*。

示例：创建 vRealize Automation 高级服务蓝图

此示例脚本会执行以下操作：

1 设置 vRealize Orchestrator 工作流用来构建服务蓝图。

2 基于工作流生成服务蓝图的内容。

3 创建服务蓝图实体。

4 发布服务蓝图。

表 4-9 输入变量

变量	类型
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);
```

示例：创建 vRealize Automation 批准策略

此示例脚本会执行以下操作：

1 获取批准策略类型。

2 设置需要批准的用户和组。

3 设置批准级别。

4 定义置备前批准阶段。

5 定义置备后批准阶段。

6 定义批准策略规范，例如名称、描述和类型。

7 创建批准策略。

8 发布批准策略。发布批准策略后，策略将变为只读。

表 4-10 输入变量

变量	类型
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