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.

vRealize Automation 用 VMware vRealize Orchestrator プラグインの概要

1

vRealize Automation 用の VMware vRealize Orchestrator プラグインは、vRealize Orchestrator と vRealize Automation 間の相互通信を可能にします。

vRealize Automation プラグインを使用して、以下の vRealize Automation 機能のワークフローを作成、実行できます。

- XaaS カスタム リソースおよびブループリントの管理
- カタログ アイテムおよびリソースの管理と要求
- 資格の構成
- ポリシー構成の承認
- 作業アイテムの相互通信
- vSphere および vCloud Director 仮想マシンのプロビジョニングとプロビジョニング後のアクション
- vRealize Automation IaaS モデル上での作成、読み込み、削除 (CRUD) の操作

vRealize Orchestrator の vRealize Automation プラグインにおける役割

Orchestrator クライアントを使用してワークフローを実行および作成し、プラグイン API にアクセスします。組み込みの vRealize Orchestrator インスタンスを、お使いの vRealize Automation インストールに使用するか、外部 vRealize Orchestrator サーバのいずれかに使用できます。

vRealize Orchestrator は、vRealize Automation プラグインを強力にします。vRealize Orchestrator は、開発およびプロセス自動化のプラットフォームであり、VMware クラウド スタックおよびサード パーティの技術を管理するための拡張ワークフローを提供します。

Orchestrator のオープンなプラグイン アーキテクチャにより、vRealize Orchestrator を各種の管理ソリューションと統合することができます。

vRealize Automation プラグインの構成 2

vRealize Automation ホストおよび IaaS ホストを追加して、プラグインを構成します。

構成ワークフロー

構成 ワークフロー カテゴリ内のワークフローを使用して、vRealize Automation ホストを管理することができます。

vRealize Automation ホスト

Orchestrator クライアントの **ワークフロー** ビューに表示されるプラグイン ライブラリの **Configuration** サブディレクトリから、これらのワークフローにアクセスすることができます。

ワークフロー名	説明
vRA ホストの追加	vRealize Automation ホストをプラグイン インベントリに追加します。テナント管理タスクおよび管理タスクの場合は、**インベントリ** ビューを使用して、各テナントに対してワークフローを実行できます。テナントに対してプラグインの全機能を使用するには、テナントごとに専用の vRealize Automation ホストを作成します。
コンポーネント レジストリを使用した vRA ホストの追加	[ユーザーごとのセッション] 接続を使用して、vRealize Automation ホストをプラグイン インベントリに追加します。vRealize Automation システム管理者の認証情報を使用して Orchestrator クライアントにログインしている必要があります。 この機能を外部の vRealize Orchestrator サーバで使用するには、Orchestrator サーバを vRealize Automation コンポーネント レジストリに登録する必要があります。
vRA ホストの IaaS ホストの追加	選択した vRealize Automation ホストの IaaS ホストをプラグイン インベントリに追加します。
vRA ホストの削除	プラグイン インベントリから vRealize Automation ホストを削除します。
vRA ホストの更新	プラグイン インベントリ内の vRealize Automation ホストを更新します。
vRA ホストの検証	vRealize Automation ホストとその接続を検証します。

注意 vRealize Orchestrator サーバが vRealize Automation コンポーネント レジストリに登録されている場合は、vRealize Automation ホストが Default という名前で自動的に追加されます。Default ホストは、[ユーザーごとのセッション] 接続を使用して、デフォルト テナントに接続します。vRealize Automation のインストール時にに組み込まれている Orchestrator サーバは、デフォルトで vRealize Automation コンポーネント レジストリに登録されます。

vRealize Automation IaaS ホスト

Orchestrator クライアントの **ワークフロー** ビューに表示されるプラグイン ライブラリの **Infrastructure Administration > Configuration** サブディレクトリから、これらのワークフローにアクセスすることができます。

vRealize Automation のインストール時に組み込まれている vRealize Orchestrator サーバは、デフォルトで vRealize Automation コンポーネント レジストリに登録されます。

ワークフロー名	説明
IaaS ホストの追加	vRealize Automation IaaS ホストをプラグイン インベントリに追加します。このワークフローは、「vRA ホストの IaaS ホストの追加」と機能性は同じですが、vRealize Automation ホストが不要である点が異なります。
IaaS ホストの削除	プラグイン インベントリから vRealize Automation IaaS ホストを削除します。
IaaS ホストの更新	プラグイン インベントリ内の vRealize Automation IaaS ホストを更新します。
IaaS ホストの検証	vRealize Automation IaaS ホストとその接続を検証します。

vRealize Automation ホストの追加

vRealize Automation ホストを追加するワークフローを実行し、ホストの接続パラメータを設定することができます。

手順

1 Orchestrator クライアントのドロップダウン メニューから、**実行** または **設計** を選択します。

2 **ワークフロー** ビューをクリックします。

3 **ライブラリ > vRealize Automation > 構成** の順に展開します。

4 **vRA ホストの追加** ワークフローを右クリックして **ワークフローの開始** の順に選択します。

5 **ホスト名** テキスト ボックスに一意のホストの名前を入力します。

6 **ホスト URL** テキスト ボックスにホストの URL アドレスを入力します。

例： *https://hostname*。

7 **テナント** テキスト ボックスにテナントの名前を入力します。

テナント用のプラグインの全機能を使用するには、テナントごとに専用の vRealize Automation ホストを作成します。

8 ユーザーへの確認なしで自動的に SSL 証明書をインストールするかどうかを選択します。

9 (オプション) vRealize Orchestrator が vRealize Automation からの接続または応答を待機する時間を設定するには、**接続タイムアウト（秒）** および **操作タイムアウト（秒）** テキスト ボックスにタイムアウト間隔を入力します。

10 ホストへの接続の種類を **セッション モード** ドロップダウン メニューから選択します。

オプション	アクション
共有セッション	vRealize Automation ユーザーの認証情報を、**認証ユーザー名** と **認証パスワード** のテキスト ボックスに入力します。
ユーザー セッション単位	現在ログインしているユーザーの認証情報を使用して接続します。vRealize Automation システム管理者の認証情報を使用して Orchestrator クライアントにログインしている必要があります。 このオプションを外部の vRealize Orchestrator サーバで使用するには、Orchestrator サーバを vRealize Automation コンポーネント レジストリに登録する必要があります。

11 **送信** をクリックします。

次に進む前に

vRealize Automation のインフラストラクチャ管理ホストを追加します。

IaaS ホストの追加

vRealize Automation ホストの IaaS ホストを追加するワークフローを実行し、接続パラメータを設定できます。

手順

1 Orchestrator クライアントのドロップダウン メニューから、**実行** または **設計** を選択します。

2 **ワークフロー** ビューをクリックします。

3 **ライブラリ > vRealize Automation > インフラストラクチャ管理 > 構成** の順に展開します。

4 **IaaS ホストの追加** を右クリックして **ワークフローの開始** の順に選択します。

5 **vCAC ホスト** ドロップダウン メニューから、IaaS ホストを構成する vRealize Automation ホストを選択します。

6 **ホスト名** テキスト ボックスに一意のホストの名前を入力します。

7 Model Manager がインストールされているマシンの URL を入力します。

例：https://*model_manager_machine.com*

8 SSL 証明書をインストールするには、**はい** を選択します。

9 Model Manager マシンへのアクセスにプロキシを使用するには、**はい** を選択します。

このオプションを選択した場合は、次のページでプロキシ ホストとプロキシ ポートを入力する必要があります。

10 **次へ** をクリックします。

11 明示的プロキシを設定している場合は、プロキシ ホストの URL とポートを入力します。

12 **次へ** をクリックします。

13 独自のタイムアウト値を設定するには、**いいえ** をクリックします。

14 (オプション) vRealize Orchestrator が vRealize Automation からの接続または応答を待機する時間を設定するには、**接続タイムアウト（秒）** および **操作タイムアウト（秒）** テキスト ボックスにタイムアウト間隔を入力します。

15 **次へ** をクリックします。

16 ホストの認証タイプを選択します。

オプション	説明
SSO	vCenter Single Sign-On を使用する場合に選択します。
NTLM	Active Directory インフラストラクチャが NTLM 認証に依存している場合にのみ、このオプションを選択して NT LAN Manager (NTLM) プロトコルベースの認証を有効にします。 このオプションには、NTLM 認証情報と認証オプションが必要です。

17 NTLM を選択した場合は、**次へ** をクリックし、ワークステーション　マシン名前と NetBIOS ドメイン名を入力します。

18 **送信** をクリックします。

vRealize Automation プラグインのワークフローの使用

3

vRealize Automation プラグインのワークフロー ライブラリには、カタログの操作、インフラストラクチャの管理、テナントやサービスの作成など、一般的なタスクに使用できるワークフローが含まれています。

カスタムの HTTP ヘッダー（vRealize Automation 固有の Tasks および Identity ヘッダーなど）を使用して、CRUD ワークフロー、プロビジョニング ワークフロー、およびプロビジョニング後のワークフローに適用できます。

この章では次のトピックについて説明します。

操作制限の削除

作成、読み込み、更新、削除の操作は、バージョン 7.0 から制限されています。以前のバージョンのワークフローでこれらの操作を使用している場合、それらの操作はバージョン 7.0 以降では機能しません。ワークフローを更新してサポートされている操作に合わせるか、必要な操作を再度有効にすることができます。

操作を再度有効にするには、有効にしたい操作を `operations.properties` ファイルから削除してください。ファイルにある操作のリストについて詳しくは、制限されている動作 (P. 14) を参照してください。

手順

1 vRealize Orchestrator のドロップダウン メニューから、**設計** を選択します。

2 **リソース** ビューをクリックします。

3 リソース階層で、**ライブラリ > VCAC > Util** を展開します。

4 バックアップを作成し、`operations.properties` ファイルを変更します。

   a `operations.properties` をクリックして **ファイルに保存** を選択します。

   b コピーをバックアップとして保存します。

   c 新しいコピーを作成し、再度有効にする操作を削除します。

   d 新しいファイルを保存します。

5 vRealize Orchestrator の既存のファイルを置き換えます。

   a vRealize Orchestrator で、**Util** フォルダを右クリックして、**リソースのインポート** をクリックします。

   b 新しいバージョンの `operations.properties` ファイルを参照して、**開く** をクリックします。

   c **一斉置換** をクリックして、変更したバージョンを保存します。

6 vRealize Orchestrator サーバを再起動します。

7 `operations.properties` ファイルを選択して、**ビューア** タブをクリックします。

8 有効にした操作がファイルに存在しないことを確認してください。

ファイルから削除した操作が、古いワークフローで機能するようになりました。

次に進む前に

新しいワークフローを作成する場合は、制限されている操作を使用しないようにしてください。

制限されている動作

`operations.properties` ファイルのコンテンツには、制限されている動作が含まれています。操作を再度有効にするには、有効にしたい操作を operations.properties ファイルから削除します。

以下のテキストは、`operations.properties` ファイルのデフォルト バージョンです。操作を再度有効にするには、操作制限の削除 (P. 13) を参照してください。

```
#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
```

vRealize Automation プラグイン インベントリの使用

インベントリ ビューを使用して、vRealize Automation オブジェクトに対してワークフローを実行できます。

インベントリ オブジェクトに使用できるワークフローを表示するには、**ツール > ユーザー環境設定 > インベントリ** の順に移動し、**インベントリでコンテキスト メニューを使用する** チェック ボックスを選択します。オプションが有効になると、Orchestrator インベントリでオブジェクトを右クリックしたときに、そのオブジェクトに使用可能なすべてのワークフローが表示されます。

vRealize Automation プラグインの管理ワークフローの使用

管理ワークフローを使用すると、vRealize Automation サービス、テナント、承認ポリシー、資格、ビジネス グループ、カタログ アイテム、アドバンスド サービス コンポーネントを管理できます。

一部のワークフローには、vRealize Automation ホストの `vCACCAFE:VCACHost` 入力パラメータが含まれています。vRealize Automation ホスト接続の構成方法によって、ユーザーがワークフローを実行する際に、ロールがどのように適用されるかが決まります。

- 接続を共有セッションとして構成している場合、ワークフローを実行するために必要なロールが、共有セッション用のユーザー アカウントに割り当てられている必要があります。
- 接続を各ユーザーのセッションとして構成している場合、vRealize Automation ユーザー インターフェイスでユーザーにロールが割り当てられるように、ワークフローを実行する各ユーザーに必要なロールが割り当てられている必要があります。

これらのワークフローは、vRealize Orchestrator クライアントの **ワークフロー** ビューで表示される **ライブラリ > vRealize Automation > Administration** サブディレクトリ内で見つけることができます。

Approval Policies サブディレクトリ内のワークフローを使用して、承認ポリシーを作成、管理できます。

表 3-1. 承認ポリシー

ワークフロー	説明
承認ポリシーの有効化	承認ポリシーを有効にします。承認ポリシーは有効にすると読み取り専用になります。
承認レベルの追加	常に必須とする承認レベルを承認ポリシーに追加します。承認者として特定のユーザーとグループを選択する必要があります。
承認ポリシーのコピー	承認ポリシーをコピーします。
承認ポリシーの作成	承認レベルや承認者を指定せずに承認ポリシーのドラフトを作成します。承認レベルを作成してポリシーの承認者を指定するには、[承認レベルの追加] ワークフローを実行します。
承認ポリシーの無効化	承認ポリシーを無効にします。承認ポリシーに関連付けられているすべての既存の資格を削除することもできます。
承認ポリシーの削除	ドラフト状態にある承認ポリシーを削除します。アクティブな承認ポリシーは読み取り専用です。

Business Groups サブディレクトリ内のワークフローを使用して、ビジネス グループおよびビジネス グループのカスタム プロパティを作成、管理できます。

表 3-2. ビジネス グループ

ワークフロー	説明
カスタム プロパティの追加	カスタム プロパティをビジネス グループに追加します。
ビジネス グループの作成	ビジネス グループを作成します。
ビジネス グループの削除	ビジネス グループを削除します。
カスタム プロパティの削除	カスタム プロパティをビジネス グループから削除します。
ビジネス グループの更新	デフォルトのマシン プリフィックス、Active directory コンテナ、ユーザー ロールなど、ビジネス グループの詳細を更新します。
カスタム プロパティの更新	ビジネス グループのカスタム プロパティを更新します。

Administration サブディレクトリには vRealize Automation 7.0 より前のバージョンで使用できる **Business Groups (Deprecated)** サブディレクトリが含まれています。メイン フォルダにある同じ名前のワークフローを使用してください。

Catalog Items サブディレクトリ内のワークフローを使用して、カタログ アイテムを管理できます。

表 3-3. カタログ アイテム

ワークフロー	説明
カタログ アイテムの有効化	カタログ アイテムを有効にします。ユーザーがカタログ アイテムを申請するには、まずカタログ アイテムを有効にしてサービスに割り当てる必要があります。
サービスへのカタログ アイテムの割り当て	カタログ アイテムをサービスに割り当てます。ユーザーがカタログ アイテムを申請するには、まずカタログ アイテムを有効にしてサービスに割り当てる必要があります。
カタログ アイテムの無効化	ユーザーが申請できなくなるように、カタログ アイテムを無効にしてサービス カタログから削除します。

Composite Blueprint サブディレクトリ内のワークフローを使用して、デザイン キャンバスで作成された複合ブループリントを管理できます。

表 3-4. 複合ブループリント

ワークフロー	説明
複合ブループリントの削除	未公開のブループリントをブループリントのデザイン リストから削除します。
複合ブループリントのインポート	YAML ファイルから複合ブループリントをインポートします。
複合ブループリントの公開	ドラフト状態にある複合ブループリントを公開します。
複合ブループリントの公開解除	公開済み複合ブループリントを公開解除します。

Content サブディレクトリ内のワークフローは廃止されました。インポートおよびエクスポート アクションを実行するには、Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。

表 3-5. コンテンツ

ワークフロー	説明
コンテンツのエクスポート（非推奨）	インポートおよびエクスポート アクションを実行するには、Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。
コンテンツのインポート（非推奨）	インポートおよびエクスポート アクションを実行するには、Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。
コンテンツの転送（非推奨）	インポートおよびエクスポート アクションを実行するには、Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。
コンテンツの検証（非推奨）	インポートおよびエクスポート アクションを実行するには、Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。

Entitlements サブディレクトリ内のワークフローを使用して、資格を作成、管理できます。

表 3-6. 資格

ワークフロー	説明
資格の有効化	資格を有効にします。
資格へのカタログ アイテムの割り当て	1 つ以上のカタログ アイテムを資格に割り当てます。このワークフローを使用して承認ポリシーを割り当てることもできます。
資格への即時アクションの割り当て	1 つ以上の即時アクションを資格に割り当てます。即時アクションで申請は行われません。
資格へのリソース アクションの割り当て	1 つ以上のリソース アクションを資格に割り当てます。このワークフローを使用して承認ポリシーを割り当てることもできます。
資格へのサービスの割り当て	1 つ以上のサービスを資格に割り当てます。このワークフローを使用して承認ポリシーを割り当てることもできます。
資格へのユーザーおよびグループの割り当て	1 人以上のユーザーまたは 1 つ以上のグループを資格に割り当てます。

表 3-6. 資格 (続き)

ワークフロー	説明
資格の作成（非推奨）	資格を作成します。[サブテナントの資格の作成] を使用します。
サブテナントの資格の作成	資格を作成します。
資格の無効化	資格を無効にします。
ユーザーおよびグループの資格からの割り当て解除	資格のユーザー リストからユーザーおよびグループを削除します。

Properties サブディレクトリ内のワークフローを使用して、プロパティ定義とプロパティ グループを管理できます。vRealize Automation のプロパティとの競合を避けるため、すべてのカスタム プロパティ名で、会社名または機能名の後ろにドットを付けたプリフィックスを使用します。

表 3-7. プロパティ定義

ワークフロー	説明
プロパティ定義の作成	カスタム プロパティを作成します。
プロパティ定義の削除	カスタム プロパティを削除します。

プロパティ グループはプロパティ定義の集合です。

表 3-8. プロパティ グループ

ワークフロー	説明
グループへのプロパティの追加	定義済みカスタム プロパティをグループに追加します。
プロパティ グループの作成	定義済みカスタム プロパティの追加先となるプロパティ グループを作成します。
プロパティ グループの削除	プロパティ グループを削除します。
グループからのプロパティの削除	定義済みカスタム プロパティをプロパティ グループから削除します。
プロパティ グループの更新	プロパティ グループの名前または説明を変更します。
グループのプロパティの更新	プロパティ グループのプロパティの名前、値、動作を変更します。

Services サブディレクトリ内のワークフローを使用して、サービスを管理できます。

表 3-9. サービス

ワークフロー	説明
サービスの有効化	サービスを有効にします。
サービスへのカタログ アイテムの割り当て	1 つ以上のカタログ アイテムをサービスに割り当てます。
サービスのコピー	サービスをコピーします。
サービスの作成	サービスを作成します。
サービスの無効化	サービスを無効にします。
サービスの削除	サービスを削除します。

Tenants サブディレクトリ内のワークフローを使用して、テナントを作成、管理できます。

ID ストア ワークフローは廃止されました。代替ワークフローは vRealize Automation の Directories Management API の変更に対応します。

表 3-10. テナント

ワークフロー	説明
管理者の追加	1 人以上のテナント管理者およびインフラストラクチャ管理者をテナントに追加します。
テナントへの ID ストアの追加	ID ストアを vRealize Automation ホストのテナントに追加します。このワークフローを実行できるのは、テナントを構成するシステム管理者に限られます。
テナントへの ID ストアの追加（非推奨）	「テナントへの ID ストアの追加」ワークフローを使用します。
vCAC ホストへの ID ストアの追加	ID ストアを vRealize Automation ホストとして構成されているテナントに追加します。このワークフローを実行できるのは、テナントに ID ストアを構成するテナント管理者に限られます。
vCAC ホストへの ID ストアの追加（非推奨）	「vCAC ホストへの ID ストアの追加」ワークフローを使用します。
テナントの作成	テナントを作成します。システム管理者の認証情報が追加されている vRealize Automation ホストを選択する必要があります。
テナントからの ID ストアの削除	ID ストアを vRealize Automation ホストのテナントから削除します。このワークフローを実行できるのは、テナントを構成するシステム管理者に限られます。
vCAC ホストからの ID ストアの削除	vRealize Automation ホストとして構成されているテナントから ID ストアを削除します。このワークフローを実行できるのは、テナントの ID ストアを構成するテナント管理者に限られます。
テナントの削除	テナントを削除します。
管理者の削除	1 人以上のテナント管理者およびインフラストラクチャ管理者をテナントから削除します。
テナントの ID ストアの更新	vRealize Automation ホストのテナントが使用している既存の ID ストアを更新します。このワークフローを実行できるのは、テナントを構成するシステム管理者に限られます。
テナントの ID ストアの更新（非推奨）	「テナントの ID ストアの更新」ワークフローを使用します。
vCAC ホストの ID ストアの更新	vRealize Automation ホストとして構成されているテナントの ID ストアを更新します。このワークフローを実行できるのは、テナントの ID ストアを構成するテナント管理者に限られます。
vCAC ホストの ID ストアの更新（非推奨）	「vCAC ホストの ID ストアの更新」ワークフローを使用します。
テナントの更新	既存のテナントの名前、説明、連絡先のメール アドレスを更新します。

Workflow Subscriptions サブディレクトリ内のワークフローを使用して、イベント ワークフロー サブスクリプションを管理できます。

表 3-11. ワークフロー サブスクリプション

ワークフロー	説明
ワークフロー サブスクリプションの削除	未公開のワークフロー サブスクリプションを削除します。このワークフローはシステム ワークフロー サブスクリプションとテナント ワークフロー サブスクリプションに適用されます。
システム ワークフロー サブスクリプションのエクスポート	システム ワークフロー サブスクリプションをエクスポートして、vRealize Orchestrator リソース要素として JSON 形式で保存します。 システム ワークフロー サブスクリプションは、システム イベントと全テナントのイベントに対処する特別なワークフロー サブスクリプションです。
テナント ワークフロー サブスクリプションのエクスポート	テナント ワークフロー サブスクリプションをエクスポートして、リソース要素として JSON 形式で保存します。 テナント固有のワークフローを実行する特別なワークフロー サブスクリプションです。
システム ワークフロー サブスクリプションのインポート	JSON ファイルからシステム ワークフロー サブスクリプションをインポートします。システム ワークフロー サブスクリプションはシステム イベントと全テナントのイベントに応じてトリガされます。
テナント ワークフロー サブスクリプションのインポート	エクスポート済みワークフロー サブスクリプションを JSON ファイルからインポートします。これらはテナント固有のワークフロー サブスクリプションです。
ワークフロー サブスクリプションの公開	ドラフトまたは未公開の状態にあるワークフロー サブスクリプションを公開します。このワークフローはシステム ワークフロー サブスクリプションとテナント ワークフロー サブスクリプションに適用されます。
システム ワークフロー サブスクリプションの登録	タイムアウト値、優先度の値を含むシステム ワークフロー サブスクリプションを作成します。
テナント ワークフロー サブスクリプションの登録	タイムアウト値、優先度の値を含め、テナント固有のワークフロー サブスクリプションを作成します。
ワークフロー サブスクリプションの公開解除	公開済みワークフロー サブスクリプションを公開解除します。このワークフローはシステム ワークフロー サブスクリプションとテナント ワークフロー サブスクリプションに適用されます。
ワークフロー サブスクリプションの更新	名前、説明、vRealize Orchestrator ワークフロー、サブスクリプション条件、タイムアウト値、ステータス値、優先度の値を変更します。イベント トピックやブロック状態を更新することはできません。

XaaS Custom Resources サブディレクトリ内のワークフローを使用して、XaaS カスタム リソースを作成、削除できます。

表 3-12. XaaS カスタム リソース

ワークフロー	説明
カスタム リソースの作成	カスタム リソースを作成します。
カスタム リソースの削除	カスタム リソースを削除します。

XaaS Resource Actions サブディレクトリ内のワークフローを使用して、XaaS リソース アクションを作成、管理できます。

表 3-13. XaaS リソース アクション

ワークフロー	説明
リソース アクションのクローン作成	既存のリソース アクションのコピーを作成します。
リソース アクションの作成	リソース アクションを作成します。
リソース アクションの削除	リソース アクションを削除します。
リソース アクションの公開	リソース アクションを公開します。
リソース アクションの公開解除	リソース アクションを公開解除します。

XaaS Resource Mappings サブディレクトリ内のワークフローを使用して、XaaS 以外のリソースへの XaaS マッピングを作成、管理できます。

表 3-14. XaaS リソース マッピング

ワークフロー	説明
リソース マッピングの作成	カタログ リソース タイプを vRealize Orchestrator タイプにマップします。
リソース マッピングの削除	リソース マッピングを削除します。
ターゲット基準の設定	リソース マッピングの可用性を決定する条件を指定します。

XaaS Server Configuration サブディレクトリ内のワークフローを使用して、ターゲットの Orchestrator インスタンスを管理できます。

表 3-15. XaaS サーバの構成

ワークフロー	説明
Orchestrator サーバ構成の更新	ポート、ホスト、ユーザー名、パスワードを含め、サーバ設定を変更します。
Orchestrator サーバ構成の検証	vRealize Orchestrator 設定が有効であることを確認します。ワークフローは構成が有効である場合は TRUE を、構成が有効でない場合は FALSE を返します。

XaaS Service Blueprints サブディレクトリ内のワークフローを使用して、XaaS ブループリントを作成、管理できます。

表 3-16. XaaS ブループリント

ワークフロー	説明
サービス ブループリントのクローン作成	サービス ブループリントのコピーを作成します。
サービス ブループリントの作成	サービス ブループリントを作成します。
サービス ブループリントの削除	サービス ブループリントを削除します。
サービス ブループリントの公開	サービス ブループリントを公開します。
サービス ブループリントの公開解除	サービス ブループリントを公開解除します。

vRealize Automation プラグインのインフラストラクチャ管理ワークフローの使用

インフラストラクチャ管理ワークフローを使用して基本操作を実行できます。拡張性パッケージを使用して vRealize Automation をカスタマイズすると、vRealize Orchestrator ワークフローをプロビジョニング プロセスの一環として呼び出したり、カスタム操作メニューを使用して呼び出したりできます。

インフラストラクチャ管理ワークフローは、Orchestrator クライアントの **ワークフロー** ビューで表示されるプラグイン ライブラリの **Infrastructure Administration** サブディレクトリ内で見つけることができます。

インフラストラクチャ管理ワークフローを使用すると、仮想マシンをプロビジョニングして、作成、読み取り、更新、削除といった基本操作を実行できます。

表 3-17. インフラストラクチャ管理

ワークフロー名	説明
仮想マシンの状態変更の待機	仮想マシン セットの状態が変更されるのを待機します。すべての仮想マシンが正常な状態にある場合、トリガが呼び出されてワークフローが正常に終了します。指定したいずれかの仮想マシンがエラー状態にあるか、存在しない場合、ワークフローは失敗します。次のいずれかを入力して、正常な状態にあるかどうかを指定する必要があります。 ■ **Requested** ■ **AwaitingApproval** ■ **RegisterMachine** ■ **BuildingMachine** ■ **AddingDisks** ■ **MachineProvisioned** ■ **MachineActivated** ■ **InstallTools**（VMware のみ） ■ **On** ■ **Off** ■ **TurningOn** ■ **TurningOff** ■ **ShuttingDown** ■ **Suspending** ■ **Resetting** ■ **Rebooting** ■ **Expired** ■ **DeactivateMachine** ■ **UnprovisionMachine** ■ **Disposing** ■ **Finalized**
IaaS モデル エンティティの作成	指定した vRealize Automation モデルのエンティティを作成して保持します。
IaaS モデル エンティティの削除	指定した vRealize Automation モデル エンティティを削除します。
プロビジョニング後のアクションの呼び出し（非推奨）	「リソース アクションの申請」ワークフローを使用します。
ブループリントからの仮想マシンのプロビジョニング（vRealize Automation 7.0 で廃止済み）	「カタログ アイテムの申請」または「プロビジョニング申請を使用したカタログ アイテムの申請」に置き換えられました。

表 3-17. インフラストラクチャ管理 (続き)

ワークフロー名	説明
カスタム フィルタを使用した IaaS エンティティの読み取り	カスタム フィルタを使用して vRealize Automation エンティティのリストを読み取ります。フィルタを指定しない場合は、すべてのエンティティが返ります。
システム クエリを使用した IaaS エンティティの読み取り	OData システム フィルタを使用して vRealize Automation エンティティのリストを読み取ります。システム フィルタは OData URI 規則に対して適用されます。
IaaS モデル エンティティの読み取り	ID を使用して vRealize Automation モデル エンティティを読み取ります。
IaaS モデル エンティティの更新	ID を使用して vRealize Automation モデル エンティティを更新します。

Extensibility サブディレクトリ内のワークフローを使用して vRealize Automation をカスタマイズし、vRealize Orchestrator ワークフローをプロビジョニング プロセスの一環として呼び出すか、カスタム操作メニューを使用して呼び出します。

サブディレクトリには、IaaS の認証情報、エンドポイント、エンタープライズ グループ、マシン プリフィックスなどのエンティティを管理するためのワークフローも含まれています。

表 3-18. 拡張性

ワークフロー名	説明
vCO カスタマイズのインストール	カスタマイズされた状態変更ワークフローやメニュー操作ワークフローを含め、Orchestrator カスタマイズをインストールします。
vCO カスタマイズのアンインストール	カスタマイズされた状態変更ワークフローやメニュー操作ワークフローを含め、Orchestrator カスタマイズをアンインストールします。
IaaS 仮想マシンの予約の変更	予約、ビジネス グループなど、管理対象仮想マシンの属性を変更します。
IaaS 仮想マシンのインポート（非推奨）	Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。
vCenter 仮想マシンのインポート（非推奨）	Cloud Client を使用します。Cloud Client のダウンロードとドキュメントについては、https://developercenter.vmware.com/tool/cloudclient にアクセスしてください。
仮想マシンの登録解除（vRealize Automation 7.0 で廃止済み）	代替ワークフローは提供されていません。
ブループリントおよびその仮想マシンへのメニュー操作の割り当て（非推奨）	仮想マシンのメニュー操作を追加または更新します。 非推奨ではない代替ワークフローには、「資格内のリソースアクションの割り当て」や、「複合ブループリントのインポート」が含まれます。
仮想マシンへのメニュー操作の割り当て（非推奨）	ID を使用して vRealize Automation モデル エンティティを更新します。 非推奨ではない代替ワークフローには、「資格内のリソースアクションの割り当て」や、「複合ブループリントのインポート」が含まれます。
ブループリントおよびその仮想マシンへの状態変更ワークフローの割り当て（非推奨）	vRealize Automation のイベント ブローカ サブスクリプションによって置き換えられました。
メニュー操作のカスタマイズ（vRealize Automation 7.0 で廃止済み）	代替ワークフローは提供されていません。

表 3-18. 拡張性 (続き)

ワークフロー名	説明
ブループリントおよびその仮想マシンからのメニュー操作の削除（vRealize Automation 7.0 で廃止済み）	代替ワークフローは提供されていません。
ブループリントおよびその仮想マシンからの状態変更ワークフローの削除	ブループリントおよびその仮想マシンから状態変更ワークフローを削除します。

vRealize Automation IaaS モデル エンティティの作成

ワークフローを実行して、単純または複雑な vRealize Automation IaaS エンティティ（仮想マシンからユーザーへの参照など）を作成することができます。

手順

1 Orchestrator クライアントのドロップダウン メニューから、**実行** または **設計** を選択します。

2 **ワークフロー** ビューをクリックします。

3 **ライブラリ > vRealize Automation > インフラストラクチャ管理** の順に展開します。

4 **IaaS モデル エンティティの作成** ワークフローを右クリックし、**ワークフローの開始** を選択します。

5 vRealize Automation ホストの順に選択します。

6 **モデル名** テキスト ボックスにモデルの名前を入力します。

7 **エンティティ セット名** テキスト ボックスにエンティティ セットの名前を入力します。

スクリプトまたは REST API を使用して、[単純なプロパティ]、[複雑なプロパティへのリンク]、および [HTTP ヘッダー] プロパティを設定します。

8 **送信** をクリックして、ワークフローを実行します。

vRealize Automation IaaS モデル エンティティの読み込み

ワークフローを実行して、vRealize Automation IaaS モデル エンティティを読み込むことができます。

手順

1 Orchestrator クライアントのドロップダウン メニューから、**実行** または **設計** を選択します。

2 **ワークフロー** ビューをクリックします。

3 **ライブラリ > vRealize Automation > インフラストラクチャ管理** の順に展開します。

4 **IaaS モデル エンティティの読み込み** ワークフローを右クリックし、**ワークフローの開始** を選択します。

5 vRealize Automation ホストの順に選択します。

6 **モデル名** テキスト ボックスにモデルの名前を入力します。

7 **エンティティ セット名** テキスト ボックスにエンティティ セットの名前を入力します。

スクリプトまたは REST API を使用して、[HTTP ヘッダー] プロパティを設定します。

8 **送信** をクリックして、ワークフローを実行します。

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.

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.

Workflow	Description
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.

vRealize Automation プラグインのサンプル ワークフローの使用

サンプル ワークフローは例として使用したり、独自のカスタム ワークフローを作成する際の開始点として使用したりできます。

これらのワークフローは、vRealize Orchestrator クライアントの **ワークフロー** ビューで表示されるプラグイン ライブラリの **Sample** サブディレクトリで見つけることができます。

ワークフロー名	説明
権限の作成	認証クライアントと通信するためのサンプル スクリプトと、vRealize Automation で権限を作成するための権限サービスを提供します。
テナントの作成	デフォルト テナントと同じ vRealize Automation ホストと Active Directory 構成を使用してテナントを作成します。このワークフローを実行するには、システム管理者の認証情報を使用して追加した vRealize Automation ホストを選択します。Active Directory 設定はワークフローを実行する前に変更できます。
カタログ アイテムの一覧表示	選択したテナントのカタログ アイテムのリストを返します。
カタログ アイテムのプロビジョニング申請の JSON 出力	カタログ アイテムのデフォルトの申請フォームを取得して、JSON 形式でコンソール ログに追加します。このデータを使用してプロビジョニング申請をカスタマイズできます。また、この情報を使用して **プロビジョニング申請を使用したカタログ アイテムの申請** ワークフローを変更できます。

vRealize Automation プラグインの API へのアクセス

Orchestrator に用意されている API Explorer を使用すると、vRealize Automation プラグインの API を検索したり、スクリプト化された要素で使用できる JavaScript オブジェクトのドキュメントを参照したりすることができます。

最新版の vRealize Automation API ドキュメントについては、
https://www.vmware.com/support/pubs/vcac-pubs.html を参照してください。

手順

1 Orchestrator クライアントに管理者としてログインします。

2 **ツール > API Explorer** を選択します。

3 左側のペインで **vCAC** モジュールと **VCACCAFE** モジュールをダブルクリックして、vRealize Automation プラグインの API オブジェクトの階層リストを展開します。

次に進む前に

API 要素からコードをコピーしてスクリプト処理ボックスに貼り付けることができます。API スクリプティングの詳細については、『*VMware vRealize Orchestrator における開発*』を参照してください。

開発のベスト プラクティスの詳細については、『vRealize Orchestrator ドキュメント』を参照してください。

サンプル vRealize Automation プラグイン スクリプト

4

JavaScript サンプルを切り取り、貼り付け、および編集することで、vRealize Automation タスク自動化用のオリジナル カスタム スクリプトを開発することができます。

この章では次のトピックについて説明します。

CRUD インフラストラクチャ管理サンプル スクリプト

JavaScript サンプルを切り取り、貼り付け、および編集することで、CRUD vRealize Automation タスクのスクリプトを記述できます。

vRealize Orchestrator でのスクリプト記述の詳細については、『*VMware vRealize Orchestrator を使用した開発*』を参照してください。

例: vRealize Automation モデル エンティティの作成

このサンプル スクリプトは、次のアクションを実行します。

1 モデル名およびエンティティの設定名を定義します。

2 ホスト プリフィックスのプロパティを定義します。

3 ホスト プリフィックスのエンティティを保存します。

4 プロビジョニング グループのプロパティを定義します。

5 プロビジョニング グループをリンクとして定義します。

6 プロビジョニング グループのエンティティをホスト名のプリフィックスを付けて保存します。

表 4-1. 入力変数

変数	タイプ
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);
```

例: vRealize Automation モデル エンティティの更新

このサンプル スクリプトは、次のアクションを実行します。

1 指定のエンティティからホスト ID を取得します。

2 指定のエンティティからモデル名を取得します。

3 指定のエンティティからエンティティの設定名を取得します。

4 指定のエンティティからエンティティ ID を取得します。

5 更新するプロパティを定義します。

6 エンティティを更新するアクションを開始します。

表 4-2. 入力変数

変数	Type
entity	vCAC:Entity
updatedDescription	文字列

```
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);
```

例: vRealize Automation モデル エンティティの読み込み

このサンプル スクリプトは、次のアクションを実行します。

1 モデル名およびエンティティの設定名を定義します。

2 プロパティ オブジェクトにブループリント ID を定義します。

3 エンティティを読み込みます。

表 4-3. 入力変数

変数	Type
host	vCAC:VcacHost
blueprintID	文字列

```
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);
```

例: vRealize Automation モデル エンティティの削除

このサンプル スクリプトは、次のアクションを実行します。

1 指定のエンティティからホスト ID を取得します。

2 指定のエンティティからモデル名を取得します。

3 指定のエンティティからエンティティの設定名を取得します。

4 指定のエンティティからエンティティ ID を取得します。

5 エンティティを削除するアクションを開始します。

表 4-4. 入力変数

変数	タイプ
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);
```

例: カスタム フィルタを使用した vRealize Automation エンティティの読み取り

このサンプル スクリプトは、次のアクションを実行します。

1 モデル名およびエンティティの設定名を定義します。

2 エンティティのフィルタとして使用するプロパティを定義します。

3 エンティティのリストを読み込みます。

表 4-5. 入力変数

変数	Type
host	vCAC:VcacHost
templateName	文字列

```
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);
```

例: システム クエリを使用した vRealize Automation エンティティの読み取り

このサンプル スクリプトは、次のアクションを実行します。

1 モデル名およびエンティティの設定名を定義します。

2 エンティティのフィルタとして使用するシステム クエリを指定し、マシン状態とコンポーネント フラグでフィルタされたすべての仮想マシンの上位 10 位の結果を選択します。

3 エンティティのリストを読み込みます。

表 4-6. 入力変数

変数	タイプ
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);
```

vRealize Automation エンティティのサンプル スクリプトの検索

JavaScript サンプルを切り取り、貼り付け、および編集することで、vCACCAFEEntitiesFinder スクリプト作成ユーティリティ オブジェクトで vRealize Automation エンティティを検索するためのスクリプトを記述できます。

vRealize Orchestrator でのスクリプト記述の詳細については、『*VMware vRealize Orchestrator を使用した開発*』を参照してください。

例: 名前フィルタによるカタログリソースの検索

表 4-7. 入力変数

変数	タイプ
host	vCACCAFE:VcacHost

次のサンプルのいずれかを使用できます。

- このサンプル スクリプトは、*name_of_the_resource* のクエリに名前および説明で一致するターゲット ホストのすべてのカタログ リソースを取得します。

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

- このサンプル スクリプトは、次のアクションを実行します。
  a コンシューマ リソース サービスを取得し、vCACCAFEPageOdataRequest オブジェクトのインスタンスを Pageable パラメータとして渡して get メソッドを開始します。

  b vCACCAFEPageOdataRequest オブジェクトを、*name_of_the_resource* 文字列に一致する name 属性の 1 つのフィルタとして OData クエリを提供して作成します。

```
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));
```

例: 所有者フィルタによるカタログリソースの検索

このサンプル スクリプトは、次のアクションを実行します。

1 コンシューマ リソース サービスを取得し、vCACCAFEPageOdataRequest オブジェクトのインスタンスを Pageable パラメータとして渡して get メソッドを開始します。

2 vCACCAFEPageOdataRequest オブジェクトを、*user@domain.com* 文字列に一致する owner/ref 属性の 1 つのフィルタとして OData クエリを提供して作成します。

   owners/ref 属性はカタログ リソースの内部構造およびフィールドに基づいた構成です。vCACCAFECatalogResource エンティティは owners 属性を持ち、これは vCACCAFECatalogPrincipal エンティティの集合です。vCACCAFECatalogPrincipal エンティティはユーザーのプリンシパル ID を表す文字列である ref プロパティを持ちます。

```
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));
```

例: 名前および所有者フィルタによるカタログリソースの検索

このサンプル スクリプトでは、前述の 2 つのサンプルの `OData` クエリを `vCACCAFEFilterParam.and(array of conditions)` 論理演算子を使用して一つの条件に結合します。

```
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));
```

その他の条件は、`vCACCAFEFilterParam.group(array of parameters)`、`vCACCAFEFilterParam.not(parameter)`、`vCACCAFEFilterParam.startsWith(id, string)`、`vCACCAFEFilterParam.endsWith(id, string)`、`vCACCAFEFilterParam.greaterThan(id, number)`、`vCACCAFEFilterParam.lessThan(id, number)` など異なる論理演算子を使用して定義できます。

vRealize Automation がプロビジョニングしたリソースを取得するためのサンプル スクリプト

JavaScript のサンプルを切り取り、貼り付け、および編集することで、vRealize Automation がプロビジョニングしたリソースの実際のエンティティを取得するスクリプトを作成できます。

vRealize Automation では `CatalogResource` タイプが、プロビジョニングされたリソースを表します。このタイプには `ProviderBinding` タイプの属性があり、以下の属性でカタログ リソースとそのプロバイダ間の関係を表します。

- `bindingId` - プロバイダに固有のエンティティの ID を表します。
- `providerRef` - vRealize Automation コンポーネント レジストリに登録されたサービスに直接対応するカタログ プロバイダを識別します。

vRealize Orchestrator でのスクリプト記述の詳細については、『*VMware vRealize Orchestrator を使用した開発*』を参照してください。

例: vRealize Automation カタログ リソースとしてプロビジョニングされた仮想マシンの取得

このサンプルでは、vRealize Automation ホストとその IaaS ホストを入力パラメータとして使用し、プロビジョニングされたリソース ID の場合は、対応する IaaS 仮想マシンを返します。スクリプト コードは、`iaas-service` プロバイダがプロビジョニングした `Virtual Machine` タイプのカタログ リソースのみを取得します。

表 4-8. 入力変数

変数	タイプ
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);
}
```

一般的なタスクのサンプル スクリプト

JavaScript サンプルの切り取り、貼り付け、編集、またはそれらをサンプルとして使用することで、一般的な vRealize Automation タスクで使用する独自のスクリプト開発への理解を深めることができます。

vRealize Orchestrator でのスクリプト記述の詳細については、『*VMware vRealize Orchestrator を使用した開発*』を参照してください。

例: vRealize Automation アドバンスド サービス ブループリントの作成

このサンプル スクリプトは、次のアクションを実行します。

1 vRealize Orchestrator ワークフローを使用して、サービス ブループリントを構築します。

2 ワークフローに基づいてサービス ブループリントのコンテンツを生成します。

3 サービス ブループリントのエンティティを作成します。

4 サービス ブループリントを発行します。

表 4-9. 入力変数

変数	タイプ
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);
```

例: vRealize Automation 承認ポリシーの作成

このサンプル スクリプトは、次のアクションを実行します。

1 承認ポリシー タイプを取得します。

2 承認を受ける必要のあるユーザーとグループを指定します。

3 承認レベルを設定します。

4 プロビジョニング前の承認段階を定義します。

5 プロビジョニング後の承認段階を定義します。

6 名前、説明、タイプといった承認ポリシーの仕様を定義します。

7 承認ポリシーを作成します。

8 承認ポリシーを公開します。一度公開されると、承認ポリシーは読み取り専用になります。

表 4-10. 入力変数

変数	タイプ
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