Cisco UCS Director REST API リリース 6.0 手順書

初版：2016 年 09 月 16 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

目 次

はじめに

対象読者

このマニュアルは、API を使用してアプリケーションを開発および拡張する専門知識を持つソフトウェア技術者を対象としています。対象となる技術者は、Cisco UCS と、関連するネットワークおよびストレージのプロトコルの知識と、JSON、XML、および Java での作業経験が必要です。

表記法

テキストのタイプ	表示
GUI 要素	タブの見出し、領域名、フィールドのラベルのような GUI 要素は、イタリック体で示しています。 ウィンドウ、ダイアログボックス、ウィザードのタイトルのようなメイン タイトルは、**ボールド体**で示しています。
マニュアルのタイトル	マニュアルのタイトルは、イタリック体で示しています。
TUI 要素	テキストベースのユーザ インターフェイスでは、システムによって表示されるテキストは、`courier` フォントで示しています。

テキストのタイプ	表示
システム出力	システムが表示するターミナル セッションおよび情報は、courier フォントで示しています。
CLI コマンド	CLI コマンドのキーワードは、**ボールド体** CLI コマンド内の変数は、イタリック体で示しています。
[ ]	角カッコの中の要素は、省略可能です。
{x | y | z}	どれか1つを選択しなければならない必須キーワードは、波カッコで囲み、縦棒で区切って示しています。
[x | y | z]	どれか1つを選択できる省略可能なキーワードは、角カッコで囲み、縦棒で区切って示しています。
string	引用符を付けない一組の文字。stringの前後には引用符を使用しません。引用符を使用すると、その引用符も含めて string とみなされます。
< >	パスワードのように出力されない文字は、山カッコで囲んで示しています。
[ ]	システム プロンプトに対するデフォルトの応答は、角カッコで囲んで示しています。
!、#	コードの先頭に感嘆符（!）またはポンド記号（#）がある場合には、コメント行であることを示します。

（注） 「注釈」です。役立つ情報や、このマニュアル以外の参照資料などを紹介しています。

注意 「要注意」の意味です。機器の損傷またはデータ損失を予防するための注意事項が記述されています。

ヒント 「問題解決に役立つ情報」です。ヒントには、トラブルシューティングや操作方法ではなく、ワンポイントアドバイスと同様に知っておくと役立つ情報が記述される場合もあります。

ワンポイント アドバイス 「時間の節約に役立つ操作」です。ここに紹介している方法で作業を行うと、時間を短縮できます。

警告　安全上の重要事項

「危険」の意味です。人身事故を予防するための注意事項が記述されています。装置の取り扱い作業を行うときは、電気回路の危険性に注意し、一般的な事故防止策に留意してください。各警告の最後に記載されているステートメント番号を基に、装置に付属の安全についての警告を参照してください。

これらの注意事項を保管しておいてください。

関連資料

『Cisco UCS DirectorDocumentation Roadmap』

Cisco UCS Director の資料の詳細なリストについては、次の URL にある 『*Cisco UCS Director Documentation Roadmap*』 を参照してください。http://www.cisco.com/en/US/docs/unified_computing/ucs/ucs-director/doc-roadmap/b_UCSDirectorDocRoadmap.html

『Cisco UCS Documentation Roadmaps』

すべての B シリーズ マニュアルの一覧については、『*Cisco UCS B-Series Servers Documentation Roadmap*』（URL：http://www.cisco.com/go/unifiedcomputing/b-series-doc）を参照してください。

すべての C シリーズ マニュアルの一覧については、http://www.cisco.com/go/unifiedcomputing/c-series-doc で入手できる『*Cisco UCS C-Series Servers Documentation Roadmap*』を参照してください。

（注）　『*Cisco UCS B-Series Servers Documentation Roadmap*』には Cisco UCS Manager および Cisco UCS Central のドキュメントのリンクが含まれています。『*Cisco UCS C-Series Servers Documentation Roadmap*』には Cisco Integrated Management Controller のドキュメントのリンクが含まれています。

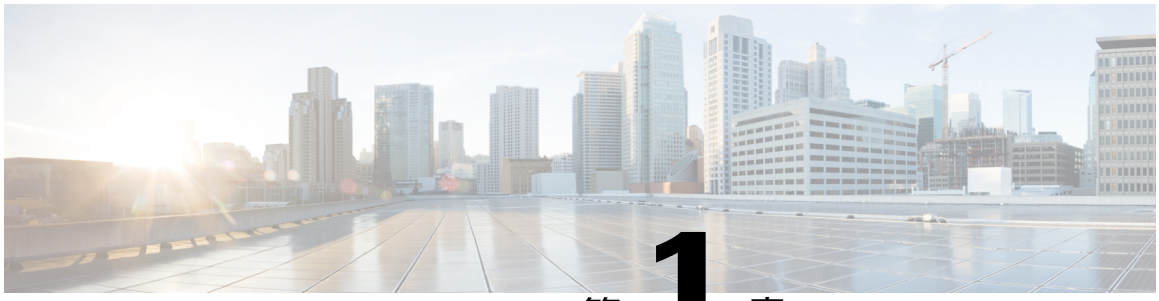
マニュアルに関するフィードバック

このマニュアルに関する技術的なフィードバック、または誤りや記載もれなどお気づきの点がございましたら、ucs-director-docfeedback@cisco.com よりコメントをお送りください。ご協力をよろしくお願いいたします。

マニュアルの入手方法およびテクニカル サポート

マニュアルの入手、Cisco Bug Search Tool（BST）の使用、サービス要求の送信、追加情報の収集の詳細については、『What's New in Cisco Product Documentation』を参照してください。

新しく作成された、または改訂されたシスコのテクニカル コンテンツをお手元に直接送信するには、『What's New in Cisco Product Documentation』RSS フィードをご購読ください。RSS フィードは無料のサービスです。



第 1 章

このリリースの新規情報および変更情報

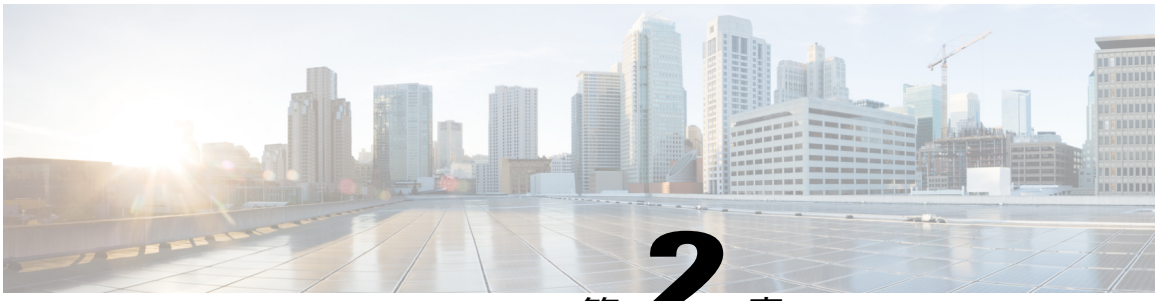
新情報および変更情報

次の表は、この最新リリースに関するガイドでの主な変更点の概要を示したものです。この表は、変更やこのリリースの新しい機能をすべて網羅するものではありません。

表 *1*：*Cisco UCS Director*、リリース *6.0* の新機能と動作変更

機能	新着情報	参照先
グループを管理する REST API のサポート	次の API が更新されました。 • userAPICreateGroup • userAPIUpdateGroup	グループの作成，（12 ページ） グループの変更，（17 ページ）
パスワードを変更する REST API のサポート	以下の API が追加されました。 • userAPIModifyLoginProfilePassword • userAPIModifyUserPassword	ログインしているユーザのパスワードのリセット，（20 ページ） ユーザのパスワードのリセット，（22 ページ）
APIC コンテナに階層を追加する REST API のサポート	次の API が追加されました。 • userAPIAddTierToContainer	APIC コンテナへの階層の追加，（71 ページ）
VM をプロビジョニングする REST API のサポート	次の API が追加されました。 • userAPIProvisionRequest	VM のプロビジョニング，（80 ページ）

機能	新着情報	参照先
データストアを管理する REST API のサポート	以下の API が追加されました。 • userAPIGetEligibleDataStoreClustersForCreateNewDisk • userAPIGetEligibleDataStoresForCreateNewDisk	適格なデータストア クラスタのリストの取得, （119 ページ） 適格なデータ ストアのリストの取得, （121 ページ）



第 2 章

概要

この章は、次の項で構成されています。

Cisco UCS Director REST API の開始

Cisco UCS Director REST API を使用すると、アプリケーションは Cisco UCS Director とプログラム的に対話できるようになります。REST API リクエストによって、Cisco UCS Director のリソースにアクセスできます。API コールを使用すると、Cisco UCS Director ワークフローを実行し、スイッチ、アダプタ、ポリシー、およびその他のハードウェアおよびソフトウェア コンポーネントの構成に変更を加えることができます。

開発環境のセットアップ方法の詳細については、『Cisco UCS Director REST API Getting Started Guide』を参照してください。

例の構造

記述表題の下で、例はそれぞれ次のセクションから構成されます。

目標

例が設計された目的。

コンテキスト

いつ例を使用するか、使用しないか、およびその理由。

前提条件

例を機能させるために必要な条件。

REST の URL

REST API を渡すための REST URL。

コンポーネント

例で使用されるオブジェクトとメソッド、表示される入力変数。

コード

例のコード。

結果

例のコードから想定される出力。

実装

例を実行するときは、変更内容を含める必要がある場合があることに注意してください。

関連項目

関連する例

例の使用方法

このマニュアルは、Cisco UCS Director Orchestrator で使用するサーバ側のスクリプト ソリューションである REST API を使用する際に役立つ例のコレクションです。手順書と同様に、少なくとも 3 つの方法でこのマニュアルを使用できます。

- 記載されているとおりに例に従うことで（もちろん使用する変数に置き換えて）、従う手順のすべてを把握していなくてもタスクを完了することができます。
- テンプレートとして例を使用し、ワーク内の同様のタスクに適合させることができます。
- 例を検討することで、REST API Javadoc の参照とともに、REST API で「どのようなことが行われるか」について把握でき、スクリプトを作成する必要があるその他のタスクでの異なるメソッドの使用について概要を把握できます。

例は、一般的な使用例を説明するために選択されています。その目的はこれらの 3 つのすべての方法で円滑に使用できるようにすることです。

例：仮想マシンのセルフサービス プロビジョニング

この例は、ユーザが仮想マシン（VM）をセルフ プロビジョニングするための単純なタスクを実行するために REST API を使用する方法を示しています。

この使用例に関係する REST API コールを要約し、要求およびシステム応答について詳しく説明します。応答は一般的に実装と一致しますが、Cisco UCS Director データベースの状態によっては完全に一致するわけではありません。必要なパラメータを応答から取得し、API 要求を通じてアプリケーションに渡す必要があります。

事前定義されたカタログ項目を使用して仮想マシン（VM）をプロビジョニングできます。これを行うには、カタログのリストを表示して、適切なサービス コンテナ カタログを選択する必要があります。選択したカタログを使用してサービス コンテナを作成し、サービス コンテナで VM をセルフプロビジョニングするためのサービス要求を送信できます。

セルフサービスの VM プロビジョニングを実行するには、次の REST API を順番に実行します。

1. UserAPIGetAllCatalogs：クラウド名とグループ名を含むカタログのリストを取得して、VM をプロビジョニングするカタログを選択します。
2. UserAPIServiceContainerCatalogRequest：選択したカタログを使用して VM をプロビジョニングするためのサービス コンテナを作成するサービス リクエストを送信します。
3. UserAPIGetServiceRequestWorkFlow：オプション。サービス リクエストのワークフローの詳細を表示します。
4. userAPISubmitServiceRequest：VM をプロビジョニングするサービス リクエストを送信します。

ステップ 1 userAPIGetAllCatalogs API を使用して、VM がバインドされているクラウド名とグループ名を含むカタログのリストを取得します。その後、返されるリストからカタログを選択できます。

要求

```
/app/api/rest?formatType=json&opName=userAPIGetAllCatalogs&opData={}
```

応答


```
{
"serviceResult":{"rows":[{"Catalog_ID":"5","Catalog_Name":"VNX_Pranita","Folder":"Advanced","Catalog_Type":"Advanced",
"Template_Name":"Not Applicable","Catalog_Description":"","Cloud":"","Image":"","Group":"Default
Group","Icon":
"/app/images/temp/1436492144835_ibm.png","OS":"","Additional_OS_Info":"","Applications":"","Additional_Application_Details":"",
"Status":"OK"},{"Catalog_ID":"6","Catalog_Name":"MSP_CAT","Folder":"Advanced","Catalog_Type":"Advanced","Template_Name":"Not
 Applicable",
"Catalog_Description":"","Cloud":"","Image":"","Group":"Default
Group","Icon":"/app/images/temp/1436492144835_ibm.png","OS":"","Additional_OS_Info":"","Applications":"",
"Additional_Application_Details":"","Status":"OK"},{"Catalog_ID":"7","Catalog_Name":"VNX_Update_Tenant","Folder":"Advanced",
"Catalog_Type":"Advanced","Template_Name":"Not
Applicable","Catalog_Description":"","Cloud":"","Image":"","Group":"Default Group",
"Icon":"/app/images/temp/1436492144835_ibm.png","OS":"","Additional_OS_Info":"","Applications":"","Additional_Application_Details":"",
"Status":"OK"},{"Catalog_ID":"8","Catalog_Name":"Pja_cat","Folder":"Advanced","Catalog_Type":"Advanced","Template_Name":"Not
 Applicable",
"Catalog_Description":"","Cloud":"","Image":"","Group":"Default
```

```
Group","Icon":"/app/images/temp/1436492144835_ibm.png",
"OS":"","Additional_OS_Info":"","Applications":"","Additional_Application_Details":"","Status":"OK"},
{"Catalog_ID":"10","Catalog_Name":"pja_upd","Folder":"Advanced","Catalog_Type":"Advanced","Template_Name":"Not
 Applicable",
"Catalog_Description":"","Cloud":"","Image":"","Group":"pja_sep","Icon":"/app/images/temp/1436492144835_ibm.png","OS":"",
"Additional_OS_Info":"","Applications":"","Additional_Application_Details":"","Status":"OK"},{"Catalog_ID":"4","Catalog_Name":"zmn_con",
"Folder":"Service Container","Catalog_Type":"Service
Container","Template_Name":"zmnACT","Catalog_Description":"","Cloud":"","Image":"",
"Group":"All
Groups","Icon":"/app/images/temp/1436514875643_container_clear_64x64.png","OS":"","Additional_OS_Info":"","Applications":"",
"Additional_Application_Details":"","Status":"OK"},{"Catalog_ID":"9","Catalog_Name":"ayc_LB","Folder":"Service
 Container","Catalog_Type":
"Service
Container","Template_Name":"aycACT","Catalog_Description":"","Cloud":"","Image":"","Group":"apREGsep9_org1","Icon":
"/app/images/temp/1436514875643_container_clear_64x64.png","OS":"","Additional_OS_Info":"",
"Applications":"","Additional_Application_Details":"","Status":"OK"},{"Catalog_ID":"11","Catalog_Name":"pja_con","Folder":
"Service Container","Catalog_Type":"Service
Container","Template_Name":"pja_tmp","Catalog_Description":"","Cloud":"","Image":"",
"Group":"pja_sep","Icon":"/app/images/temp/1436514875643_container_clear_64x64.png","OS":"","Additional_OS_Info":"","Applications":"",
"Additional_Application_Details":"","Status":"OK"},{"Catalog_ID":"12","Catalog_Name":"prsConCat","Folder":"Service
 Container",
"Catalog_Type":"Service
Container","Template_Name":"prsACT","Catalog_Description":"","Cloud":"","Image":"","Group":"arpAPIReg2_Org",
"Icon":"/app/images/temp/1436514875643_container_clear_64x64.png","OS":"","Additional_OS_Info":"","Applications":"",
"Additional_Application_Details":"","Status":"OK"}],"columnMetaData":null,"reportParams":{}},
"serviceError":null, "serviceName":"InfraMgr",
 "opName":"userAPIGetAllCatalogs" }
```


ステップ 2 VM をプロビジョニングし userAPIServiceContainerCatalogRequest API を使用してサービス リクエストを送信するためのサービス コンテナの作成に使用されるカタログ（pja_con など）を選択し、選択したカタログを使用してサービス コンテナを作成します。
この例では、SCN_Name と呼ばれるサービス コンテナが pja_con カタログを使用して作成されます。

要求


```
/app/api/rest?formatType=json&opName=userAPIServiceContainerCatalogRequest&opData={param0:
{"catalogName":"pja_con","groupName":"Default
Group","serviceContainerName":"SCN_Name","apiResourceLimits":
null,"networkThroughput":"1G","enableNetworkMgmt":true,"customTierLabels":[{"name":"web","value":"web"}],
"comments":"test"}}
```


応答


```
{"serviceResult":728, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPIServiceContainerCatalogRequest"}
```


URL がサービス リクエスト ID を返します。サービス コンテナを作成するためのサービス リクエスト ID は 728 です。

ステップ 3 （任意） サービス リクエストの作成後、userAPIGetServiceRequestWorkFlow API を使用してサービス要求に関する詳細および関連ワークフローの手順を取得します。SR ID（728）は、ステップ 2 の応答から渡されます。

要求


```
/app/api/rest?formatType=json&opName=userAPIGetServiceRequestWorkFlow&opData={param0:728}
```


応答


```
{
"serviceResult":{"requestId":728,"workflowCreated":1442274648591,"submittedTime":1442274648981,"cancelledTime":-1,
"cancelledByUser":null,"adminStatus":1,"executionStatus":2,"futureStartTime":1442274648591,"entries":[{"stepId":
"Initiated by
aks","executionStatus":3,"statusMessage":null,"handlerId":4,"startedTime":-1,"completedTime":1442274649606,
"validTill":-1,"startAfter":-1},{"stepId":"GetResourceRequirementFromThroughput","executionStatus":3,"statusMessage":"",
"handlerId":12,"startedTime":-1,"completedTime":1442274657097,"validTill":-1,"startAfter":-1},{"stepId":"Allocate
 APIC Container
Resources","executionStatus":2,"statusMessage":"Execution of the task resulted in
errors","handlerId":12,"startedTime":-1,
"completedTime":1442274662674,"validTill":-1,"startAfter":-1},{"stepId":"Verify Container Resource
 Limits","executionStatus":0,
```

"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"If Else",
"executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"APIC Reterive Secondary
Container","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,
"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Trigger APIC Container - DR
Site","executionStatus":0,"statusMessage":null,
"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Create Tenant Application Profile",
"executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Create Private Network
","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,
"validTill":-1,"startAfter":-1},{"stepId":"Trigger Multiple Container Tier
Creation","executionStatus":0,"statusMessage":null,
"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Wait For Service Requests",
"executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Setup APIC Container Network
Connection","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,
"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Create APIC Container
Contracts","executionStatus":0,"statusMessage":null,
"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Child workflow
(APIC Container Attached L4L7
Configuration)","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,
"validTill":-1,"startAfter":-1},{"stepId":"Provision APIC Container
VMs","executionStatus":0,"statusMessage":null,"handlerId":12,
"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Re-Sync Container
VMs","executionStatus":0,"statusMessage":
null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"If Else","executionStatus":0,
"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Wait For Service

Requests","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},
{"stepId":"Child workflow
(APICContainerSRMSettings)","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,
"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Initiate APIC Container BM
Provisioning","executionStatus":0,"statusMessage":
null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":"Send Container Email","executionStatus":
0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,"startAfter":-1},{"stepId":
"GetMSPAdminEmailAddresses","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":-1,"validTill":-1,
"startAfter":-1},{"stepId":"Send Container
Email","executionStatus":0,"statusMessage":null,"handlerId":12,"startedTime":-1,"completedTime":
-1,"validTill":-1,"startAfter":-1},{"stepId":"Complete","executionStatus":0,"statusMessage":null,"handlerId":13,"startedTime":-1,
"completedTime":-1,"validTill":-1,"startAfter":-1}]}, "serviceError":null, "serviceName":"InfraMgr", "opName":
"userAPIGetServiceRequestWorkFlow" }

応答で、stepId はワークフローによって実行されるタスクを表します。ワークフローの実行が正常に完了すると、stepId は [完了（Complete）] として表されます。SCN_Name と呼ばれるサービス コンテナが作成されます。

ステップ 4 VM をプロビジョニングするには、userAPISubmitServiceRequest API を使用してサービス リクエストを実行します。

要求

```
/app/api/rest?formatType=json&opName=userAPISubmitServiceRequest&opData={param0:"cat82",param1:"vdc82",
param2:1,param3:-1,param4:1,param5:"vm provisioning"}
```

それぞれの説明は次のとおりです。

- param0：VM のプロビジョニングに使用されるカタログの名前。
- param1：VM をプロビジョニングする必要がある仮想データセンター（VDC）の名前。

- param2：時間単位の VM のプロビジョニングの期間。設定した期間の経過後、VM は自動的にプロビジョニング解除されます。期間を無限に設定するには -1 を使用します。
- param3：これは任意のパラメータです。VM のプロビジョニングを開始する時間をスケジュールします。たとえば、「January 1, 2014, 00:00:00 GMT」と設定します。VM プロビジョニングを即時に開始するには 0 または -1 を使用します。
- param4：プロビジョニングされる VM の数。
- param5：これは任意のパラメータです。VM のプロビジョニングに関するコメント。

応答

```
{ "serviceResult":456, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPISubmitServiceRequest" }
```

VM をプロビジョニングするためのサービス リクエスト ID は 456 です。

例：プロビジョニングされた VM のロールバック

プロビジョニングされた VM が不要になった場合、ロールバック ワークフローを使用して、その VM に割り当てられたリソースの解放と再割り当てを実行できます。Write - Group Service Request ユーザ権限を持つシステム管理者またはエンドユーザが、ワークフローのロールバックを実行できます。

ロールバック ワークフローには、userAPIRollbackWorkflow API 内で VM のプロビジョンに使用されたサービス リクエストの ID を渡すタスクを含める必要があります。進行中のサービス リクエストの ID は、Cisco UCS Director（[組織（Organizations）] > [サービスリクエスト（Service Requests）]）から入手できます。

他のユーザによってプロビジョニングされた VM をロールバックする場合、そのユーザから承認を得るためのロールバック ワークフロー承認が起動されます。ロールバック ワークフローは、承認の受信後に完了します。

要求

次の REST URL は ID 456 のサービス リクエストをロールバックします。

```
/app/api/rest?formatType=json&opName=userAPIRollbackWorkflow&opData={param0:456}
```

応答

```
{ "serviceResult":458, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPIRollbackWorkflow" }
```

userAPIGetServiceRequestWorkFlow API を使用して、以下のようにロールバック ワークフローのステータスを確認します。

要求

```
/app/api/rest?formatType=json&opName=userAPIGetServiceRequestWorkFlow&opData={param0:458}
```

ロールバック ワークフローの実行が正常に完了すると、stepId は [完了（Complete）] として表されます。VM に割り当てられているリソースが解放され、再割り当てできるようになります。

例外

ワークフローのロールバックに失敗した場合、userAPIRollbackWorkflow API は例外をスローします。

まだ進行中であるサービス リクエスト ID をロールバックしようとした場合、userAPIRollbackWorkflow API は次の例外をスローします。

```
REMOTE_SERVICE_EXCEPTION: Cannot rollback work-flow for SR ID 332, when the work-flow
execution is in progress
```

すでにロールバックされたサービス リクエスト ID をロールバックしようとした場合、userAPIRollbackWorkflow API は次の例外をスローします。

```
REMOTE_SERVICE_EXCEPTION: Cannot Rollback SR:332 as it is already rolled back.
```

第 3 章

例

この章は、次の項で構成されています。

グループの管理

グループの作成

目標

Cisco UCS Directorに指定されたデータ（グループ名、説明、連絡先の詳細）で新しいグループを作成します。

コンテキスト

サービス エンドユーザまたはグループ管理者を Cisco UCS Director に追加する前にグループを作成する必要があります。サービス エンドユーザまたはグループ管理者はグループに割り当てられます。グループ内のユーザは、権限に基づいて、リソースに対する読み取り/書き込み権限を継承します。

前提条件

REST API は管理者のユーザ ID を使用して呼び出す必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPICreateGroup&opData=
{param0:{"groupName":"SDK_Demo_Group","description":"testgroup", "parentGroupId":-1,
"parentGroupName":"any","emailAddress":"sdk@cisco.com", "lastName":"adwin",
"firstName":"Michle","phoneNumber":"2344566","address":"SanJose", "groupType":0,
"enableBudget":true}}
```

応答

```
{ "serviceResult":2, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"userAPICreateGroup" }
```

コンポーネント

userAPICreateGroup API のパラメータは次のとおりです。

- String groupName：グループまたは組織の名前。
- String description：オプション。グループまたは組織の説明。
- int parentGroupId：オプション。顧客グループをマップする必要がある親グループの ID。Cisco UCS Director で groupType が 0 に設定され、MSP モードが有効にされている場合、このパラメータは必須です。
- int parentGroupName：オプション。顧客グループをマップする必要がある親グループの名前。Cisco UCS Director で groupType が 0 に設定され、MSP モードが有効にされている場合、このパラメータは必須です。

（注） 親グループを顧客グループに割り当てるには、parentGroupId パラメータと parentGroupName パラメータの両方が必須です。

- String emailAddress：この電子メール アドレスは、必要に応じて、サービス リクエストおよびリクエスト承認のステータスをグループ所有者に通知する目的で使用されます。
- String lastName：オプション。グループ所有者の姓。
- String firstName：オプション。グループ所有者の名。
- String phoneNumber：オプション。グループ所有者の電話番号。
- String address：オプション。グループ所有者の住所。
- int groupType：デフォルトでは、グループ タイプは 0 に設定されます。グループ タイプは、ユーザ グループおよび顧客組織に対しては 0、マネージド サービス プロバイダー（MSP）組織ユーザに対しては 1 に設定します。
- boolean enableBudget：オプション。boolean enableBudget：予算監視を使用するグループを作成する場合は、true に設定します。

コード

JSON ベースの API

```
import com.cisco.cuic.api.client.APIGroup;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;

public class TestuserAPICreateGroup {
 public static void main(String[] args) throws Exception {
  CuicServer server = CuicServer.getAPI("10.10.110.222",
"6F0063A7F7654561A790EACCE1E8626F", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  int groupId;
  APIGroup apiGroup = new  APIGroup();
  apiGroup.setGroupName("custom_group_2"); //Mandatory
  apiGroup.setEmailAddress("cugroup@cisco.com"); //Mandatory
  apiGroup.setGroupType(0); //Optional- can accept either 0 or 1; by default 0.
  apiGroup.setParentGroupId(2); //Optional
  apiGroup.setParentGroupName("MSP ORG 1"); //Optional
  apiGroup.setFirstName("John"); //Optional
  apiGroup.setLastName("Carry"); //Optional
  apiGroup.setAddress("City-204"); //Optional
  apiGroup.setDescription("This is custom_group_2"); //Optional
  apiGroup.setPhoneNumber("123456789"); //Optional

  groupId = instance.userAPICreateGroup(apiGroup);
  System.out.println("Group Id for the group created: "+groupId);
}
}
```

または、次のいずれかの XML ベースの API を使用できます。

- group@CREATE：顧客グループを作成する場合に使用します。
- msporg@CREATE：MSP グループを作成する場合に使用します。

group@CREATE API には次の追加のパラメータが含まれています。

- GroupCode：グループの短い名前またはコード名。この名前は、VM とホスト名テンプレートで使用されます。
- GroupSharePolicyId：このグループのユーザのグループ共有ポリシーの ID。
- allowPrivateUsers：ユーザへのリソース割り当てを許可します。

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.accounts.AddGroupConfig;
import com.cisco.cuic.api.models.accounts.AddGroupConfigResponse;

public class TestgroupCreateXMLAPI {
 public static void main(String[] args) throws Exception {
  CuicServer server = CuicServer.getAPI("172.29.110.222",
"6F0063A7F7654561A790EACCE1E8626F", "https", 443);
  AddGroupConfig instance = new AddGroupConfig(server);
  instance.setGroupName("custom_group_5");//Mandatory
  instance.setGroupDescription("This is custom_group_5"); //Optional
  instance.setParentGroup("2"); //Optional
  instance.setGroupCode("cug_5"); //Optional
  instance.setFirstName("John"); //Optional
  instance.setLastName("Kerry"); //Optional
  instance.setGroupContact("cug_5@cisco.com");//Mandatory
  instance.setAddress("City-401"); //Optional
  instance.setAllowPrivateUsers(true); //Optional
  instance.setGroupSharePolicyId("group_share_policy_1"); //Optional
  instance.setPhone("1234567"); //Optional
  AddGroupConfigResponse groupReponse = instance.execute();
  System.out.println("Group Id for the group created:
"+groupReponse.getOUTPUT_GROUP_ID());
 }
```

```
}
```

結果

Cisco UCS Director サーバでグループが正常に作成された後、一意のグループ ID が返されます。

実装

userAPICreateGroup API を使用してグループを作成できます。XML API を使用してグループを作成する場合、顧客グループを作成するには group@CREATE API を使用し、MSP グループを作成するには msporg@CREATE を使用します。

関連項目

すべてのグループのリスト化

グループの変更

グループの削除

すべてのグループのリスト化

目標

Cisco UCS Director 内のすべてのグループをリストとして取得します。

コンテキスト

Cisco UCS Director 内のグループのリストを取得します。

前提条件

要求ユーザは、グループのリストを取得する権限を持っている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIGetGroups&opData={}
```

コンポーネント

なし

コード

```
public class userAPIGetGroupsExample
 {
  public static void main(String[] args) throws Exception
  {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  List<APIGroup> groupsList = instance.userAPIGetGroups();
  for (Iterator iterator = groupsList.iterator(); iterator.hasNext();)
   {
   APIGroup apiGroup = (APIGroup) iterator.next();
   System.out.println("Group id = "+apiGroup.getGroupId());
   System.out.println(" Group Name = "+apiGroup.getGroupName());
   }
  }
}
```

結果

コードが Cisco UCS Director 内のグループのリストを返します。

実装

実装は必要ありません。

関連項目

グループの作成

グループの変更

グループの削除

グループの変更

目標

特定の詳細でグループを更新します。グループの ID、名前、タイプを更新することはできません。

コンテキスト

グループの連絡先情報、説明、親グループを更新します。

前提条件

- ログイン ユーザには、グループを変更する権限が必要です。
- 更新するグループが Cisco UCS Director で使用可能である必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIUpdateGroup&opData={param0:
{"groupId":2","groupName":"SDKTest","description":"testing","parentGroupId":9,
"parentGroupName":"Test", "emailAddress":"test@cisco.com","lastName":"","firstName":"",
"phoneNumber":"","address":"","groupType":0}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"userAPIUpdateGroup" }
```

コンポーネント

グループを更新するには、グループ名と電子メール アドレスを渡す必要があります。

- String groupName：グループまたは組織の名前。
- String description：オプション。グループまたは組織の説明。
- int parentGroupId：オプション。顧客グループをマップする必要がある親グループの ID。Cisco UCS Director で groupType が 0 に設定され、MSP モードが有効にされている場合、このパラメータは必須です。
- int parentGroupName：オプション。顧客グループをマップする必要がある親グループの名前。Cisco UCS Director で groupType が 0 に設定され、MSP モードが有効にされている場合、このパラメータは必須です。

（注） 親グループを顧客グループに割り当てるには、parentGroupId パラメータと parentGroupName パラメータの両方が必須です。

- String emailAddress：この電子メール アドレスは、必要に応じて、サービス リクエストおよびリクエスト承認のステータスをグループ所有者に通知する目的で使用されます。
- String lastName：オプション。グループ所有者の姓。
- String firstName：オプション。グループ所有者の名。
- String phoneNumber：オプション。グループ所有者の電話番号。
- String address：オプション。グループ所有者の住所。
- int groupType：デフォルトでは、グループ タイプは 0 に設定されます。グループ タイプは、ユーザ グループおよび顧客組織に対しては 0、マネージド サービス プロバイダー（MSP）組織ユーザに対しては 1 に設定します。
- boolean enableBudget：オプション。boolean enableBudget：予算監視を使用するグループを作成する場合は、true に設定します。

コード

```
import com.cisco.cuic.api.client.APIGroup;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;

public class TestuserAPIUpdateGroup {
 public static void main(String[] args)throws Exception {
  CuicServer server = CuicServer.getAPI("172.29.110.222",
"6F0063A7F7654561A790EACCE1E8626F", "https", 443);
  boolean isGroupUpdated;
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APIGroup apiGroup = new  APIGroup();
  apiGroup.setGroupName("custom_group_2"); //Mandatory
  apiGroup.setEmailAddress("cugroup@cisco.com"); //Mandatory
  apiGroup.setGroupType(0); //Optional- can accept either 0 or 1; by default 0.
  apiGroup.setParentGroupId(11); //Optional
  apiGroup.setParentGroupName("MSP ORG 2"); //Optional
  apiGroup.setFirstName("John"); //Optional
  apiGroup.setLastName("Carry"); //Optional
  apiGroup.setAddress("City-204"); //Optional
  apiGroup.setDescription("This is custom_group_2"); //Optional
  apiGroup.setPhoneNumber("123456789"); //Optional
  isGroupUpdated = instance.userAPIUpdateGroup(apiGroup);
  System.out.println("Is the Group updated: "+isGroupUpdated);
 }
}
```

結果

グループが正常に更新された場合、結果は true となります。

実装

更新する必要があるグループ名を渡して、必要なグループ プロパティを設定します。

関連項目

グループの作成

すべてのグループのリスト化

グループの削除

グループの削除

目標

特定のグループ ID に基づいて Cisco UCS Director からグループを削除します。

コンテキスト

グループに属しているユーザが Cisco UCS Director サーバのリソースにアクセスしないことを確認します。

前提条件

グループ ID が Cisco UCS Director で使用できる必要があります。userAPIGetGroups API を使用してグループ ID を取得できます。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIDeleteGroup&opData={param0:2}
```

コンポーネント

int groupId：削除する必要があるグループの一意の ID。

コード

```
public class userAPIDeleteGroupExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  //Get the Group id by executing List<APIGroup> groupsList =
instance.userAPIGetGroups();
  boolean obj = instance.userAPIDeleteGroup(2);
  System.out.println("is Deleted successfully ? " + obj);
  }
}
```

結果

グループが正常に削除された場合、結果は true となります。

実装

グループ ID を渡すことで、グループを削除します。

関連項目

グループの作成

すべてのグループのリスト化

グループの変更

ユーザの管理

ログインしているユーザのパスワードのリセット

目標

現在ログインしているユーザのパスワードをリセットします。

コンテキスト

userAPIModifyLoginProfilePassword API を使用して、現在ログインしているユーザのパスワードをリセットできます。userAPIResetMyPassword API はされているため、userAPIModifyLoginProfilePassword API を使用することを推奨します。

前提条件

oldPassword が正しく設定されている必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
userAPIModifyLoginProfilePassword&opData={param0:{"oldPassword":"apadmin1",
"newPassword":"apadmin2"}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"userAPIModifyLoginProfilePassword" }
```

コンポーネント

- String oldPassword：現在ログインしているユーザの現在のパスワード。
- String newPassword：現在ログインしているユーザの新しいパスワード。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.models.accounts.LoginProfileChangePasswordInfo;
public class TestUserAPIModifyLoginProfilePassword {
 public static void main(String[] args) {
  CuicServer server = CuicServer.getAPI("172.22.234.172",
"50E67524B33F404C80BB8E90D276A560", "https", 443, 12000, "none", "");
  UserAPIGlobal instance = new UserAPIGlobal(server);
  LoginProfileChangePasswordInfo passwordInfo = new LoginProfileChangePasswordInfo();

  passwordInfo.setOldPassword("admin1");
  passwordInfo.setNewPassword("admin2");
  boolean passwordChanged=false;
  try {
   passwordChanged = instance.userAPIModifyLoginProfilePassword(passwordInfo);
   System.out.println("Is Password changed: "+passwordChanged);
  } catch (Exception e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
  }
 }
}
```

結果

パスワードが正常にリセットされた場合、結果は true となります。

実装

現在ログインしているユーザのパスワードをリセットするには、userAPIModifyLoginProfilePassword API を呼び出して、oldPassword および newPassword を渡します。

関連項目

ユーザのパスワードのリセット

ユーザのパスワードのリセット

目標

ユーザのパスワードをリセットし、ユーザの REST API アクセス キーを再生成します。

コンテキスト

管理者ユーザは、userAPIModifyUserPassword API を使用してユーザのパスワードをリセットできます。resetAPIKey パラメータが true に設定されている場合、ユーザの REST API アクセス キーも再生成されます。userAPIResetUserPassword API は廃止となっているため、userAPIModifyUserPassword API を使用することを推奨します。

前提条件

パスワードをリセットする対象のユーザは、有効な Cisco UCS Director ユーザでなければなりません。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIModifyUserPassword&opData=
{param0:{"loginUserpassword":"admin5","loginName":"senduser","newPassword":"senduser3",
"resetAPIKey":true}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"userAPIModifyUserPassword" }
```

コンポーネント

- String loginUserpassword：リセットする必要があるユーザのパスワード。
- String loginName：パスワードをリセットする対象のユーザの名前。
- String newPassword：ユーザの新しいパスワード。
- boolean resetAPIKey：パスワードのリセット後に REST API アクセス キーを再生成する場合は、true に設定します。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.models.accounts.ModifyUserPasswordInfo;

public class TestUserAPIModifyUserPassword {

 public static void main(String[] args) {
  CuicServer server = CuicServer.getAPI("172.22.234.172",
"91E7A134048F45FD88A29FCBBFD2C233", "https", 443, 12000, "none", "");
  UserAPIGlobal instance = new UserAPIGlobal(server);
  boolean isPasswordchanged = false;
  ModifyUserPasswordInfo userPasswordInfo = new ModifyUserPasswordInfo();
  userPasswordInfo.setLoginUserpassword("admin5");
  userPasswordInfo.setLoginName("senduser");
  userPasswordInfo.setNewPassword("senduser11");
  userPasswordInfo.setResetAPIKey(true);
  try {
   isPasswordchanged = instance.userAPIModifyUserPassword(userPasswordInfo);
   System.out.println("Is password changed successfully? : "+isPasswordchanged);

  } catch (Exception e) {

   e.printStackTrace();
  }
  }

}
```

結果

パスワードが正常にリセットされた場合、結果は true となります。

実装

ユーザのパスワードをリセットするには、userAPIModifyUserPassword API を呼び出して、コンポーネントに記載されているパラメータを設定します。

関連項目

ログインしているユーザのパスワードのリセット

カタログの管理

カタログ項目の作成

目標

以下のカタログ タイプのいずれかで、VM をプロビジョニングするためのカタログ項目を作成します。

- 標準：クラウド リストにあるイメージを使用して VM プロビジョニングのためのカタログを作成する場合に使用します。
- 詳細：カタログ項目などオーケストレーション ワークフローを公開する場合に使用します。
- サービス コンテナ：カタログ項目としてアプリケーション コンテナを公開する場合に使用します。
- 仮想デスクトップ インフラストラクチャ（VDI ）：カタログ項目として Xen Desktop を公開する場合に使用します。

カタログ項目では、VM をバインドするクラウドの名前およびグループの名前などのパラメータを定義します。

コンテキスト

システム管理者がカタログ項目を作成します。

前提条件

- クラウド、イメージおよびグループが Cisco UCS Director に存在する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPICreateCatalogItem&opData=
{param0:{"catalogType":"Standard","catalogItemName":"standardCatalog1",
"catalogItemDescription":"Api crated it","catalogIcon":"VM: SUSE Linux",
"isAppliedToAllGroups":false,"groups":"Default Group","publishToEndUsers":true,
"folderName":"Standard","standardCatalog":{"cloudName":"vmware117","image":"CentOSTiny",
"category":"Generic VM","supportEmail":"sdk@cisco.com","os":"Windows Server 2012",
"otherOS":"Linux Server","appLists":"Apache Web Server","otherApps":
"Glass Fish web server","applicationCode":"sdk","credentialOption":"Do not share",
"userId":"admin","password":"root","isAutomaticGuestCustomization":false,
"enablePostProvisioningCustomActions":false,"workflowName":"Print Number",
"parameters":[{"name":"Name1","value":"Value1"},{"name":" Name2","value":" Value2"}]},
"advancedCatalog":{"workflowName":"Create Vdc","parameters":[{"name":"Name1","value":"Value1"},
{"name":"Name2","value":"Value2"}]},"ServiceContainerCatalog":{"serviceContainerTemplateName":
"servicecontainertemp1","parameters":[{"name":"Name1","value":"Vlue1"},
{"name":"Name2","value":"Value2"}]},"vdiCatalog":{"cloudName":"Cloud1","imageId":"image1",
"xenDesktopCatalog":"catalog1","category":"category1","supportEmail":"sdk@cisco.com",
"allowEndUserToOverrideCategory":true,"parameters":[]}}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPICreateCatalogItem" }
```

コンポーネント

APICatalogItem 項目

コード

```
import java.util.ArrayList;
import java.util.List;

import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.models.catalog.APICatalogItem;
import com.cisco.cuic.api.models.catalog.StandardCatalogParameters;
import com.cisco.cuic.api.models.catalog.NameValuePair;

public class UserAPICreateCatalogItemSdkSample
{
public static void main(String[] args)
{
 CuicServer server = CuicServer.getAPI("172.29.110.194",
"5AD45F2DC5ED441A9A743F1B219CC302", "http", 80);
 UserAPIGlobal instance = new UserAPIGlobal(server);
 List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>();
     StandardCatalogParameters standardCatalogParameters = new
StandardCatalogParameters();
     standardCatalogParameters.setCloudName("vmware117");
     standardCatalogParameters.setImage("CentOSTiny");
     standardCatalogParameters.setCategory("Generic VM");
     standardCatalogParameters.setSupportEmail("sdk@cisco.com");
     standardCatalogParameters.setOs("Windows Server 2012");
     standardCatalogParameters.setOtherOS("Linux Server");
     standardCatalogParameters.setAppLists("Apache Web Server");
     standardCatalogParameters.setOtherApps("Glass Fish web server");
     standardCatalogParameters.setApplicationCode("sdk");
     standardCatalogParameters.setCredentialOption("Do not share");
     standardCatalogParameters.setUserId("admin");
     standardCatalogParameters.setPassword("root");
     standardCatalogParameters.setAutomaticGuestCustomization(false);
     standardCatalogParameters.setEnablePostProvisioningCustomActions(false);
     standardCatalogParameters.setWorkflowName("Print Number");
     standardCatalogParameters.setParameters(nameValuePairs);
 APICatalogItem apiCatalogItem = new APICatalogItem();
 apiCatalogItem.setCatalogType("Standard");
 apiCatalogItem.setCatalogItemName("standardCatalog2");
 apiCatalogItem.setCatalogItemDescription("created through client code");
 apiCatalogItem.setCatalogIcon("VM: SUSE Linux");
 apiCatalogItem.setAppliedToAllGroups(false);
 apiCatalogItem.setGroups("Default Group");
 apiCatalogItem.setPublishToEndUsers(true);
 apiCatalogItem.setFolderName("Standard");
 apiCatalogItem.setStandardCatalog(standardCatalogParameters);
 boolean isCatalogItemCreated = false;
 try {
  isCatalogItemCreated = instance.userAPICreateCatalogItem(apiCatalogItem);
 } catch (Exception e) {
  System.out.error(e.getMessage());
  System.out.error("Exception occured while creating a catalog.");
 }
 System.out.println("Is Catalog Item Got Created ?"+isCatalogItemCreated);
}
}
```

結果

カタログ項目が正常に作成された場合、結果は true となります。

実装

必要な情報を渡すことで APICatalogItem を作成し、userAPICreateCatalogItem API を呼び出してカタログ項目を作成します。

関連項目

カタログの詳細の取得

カタログ項目の削除

カタログの詳細の取得

目標

カタログ名を使用してカタログの詳細を取得します。

コンテキスト

カタログの詳細は、システム管理者またはエンド ユーザによって取得されます。

前提条件

- 要求されたカタログ項目が存在する必要があります。
- カタログ名が分かっている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIGetAllCatalogs&opData={}
```

コンポーネント

なし

コード

```
public class userAPIGetCatalogDetailsExampe
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APIProvisionParams params = instance.userAPIGetCatalogDetails("SDKCat");
  System.out.println(" Catalog name "+params.getCatalogName());
  System.out.println(" vDC name "+params.getVdcName());
  }
 }
```

結果

カタログの詳細が正常に取得された場合、結果は true となります。

実装

カタログの詳細を取得するには、カタログ名を渡して userAPIGetCatalogDetails API を呼び出します。

関連項目

カタログ項目の作成

カタログ項目の削除

カタログ項目の削除

目標

カタログ名を渡して、カタログ項目を削除します。

コンテキスト

カタログ項目はシステム管理者が削除できます。

前提条件

削除するカタログ項目が存在する必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIDeleteCatalogItem&opData={param0:"sdkCatalog"}
```

コンポーネント

catalogItemName：削除する必要のあるカタログ項目の名前。

コード

```
public class userAPIDeleteCatalogItemExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  boolean isDeleted = instance.userAPIDeleteCatalogItem("SDKCat");
  System.out.println("is Deleted ?"+isDeleted);
 }
}
```

結果

カタログ項目が正常に削除された場合、結果は true となります。

実装

既存のカタログ項目を削除するには、カタログ名を渡して userAPIDeleteCatalogItem API を呼び出します。

関連項目

カタログ項目の作成

カタログの詳細の取得

物理アカウントの管理

物理アカウントの作成

目標

Cisco UCS Director で物理的資産のアカウントを作成します。XML ベースの API を使用してアカウントを作成する場合は、有効な値を渡すために、[コンポーネント（Components）]セクションのアカウント カテゴリおよびアカウント タイプ パラメータの使用可能な値を参照してください。

コンテキスト

物理インフラストラクチャ リソースを管理します。

前提条件

- サイトおよびポッドが存在し到達可能である必要があります。
- 物理アカウントのリソースに到達可能である必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=accounts:userAPICreateInfraAccount&opData={param0:
{"accountName":"Test Vmware82","podName":"Default Pod","accountCategory":-1,"accountType":
"VMWare","deviceCategory":"","description":"sample","contact":"sample","destinationIPAddress":
"172.29.109.82","login":"administrator","password":"cloupia123","enablePassword":"","protocol":
"https","port":443,"infraAccountSupportDetailsInfo":null}}
```

コンポーネント

• InfraAccountDetails infraAccountDetails

InfraAccountDetails には、次のパラメータが含まれます。

◦ String accountName：アカウントに割り当てる一意の名前。

◦ String podName：アカウントが属するポッド。

◦ int accountCategory：アカウントのカテゴリ。アカウント カテゴリの使用可能な値は次のとおりです。

整数	アカウント カテゴリ
1	コンピューティング
2	ストレージ
3	ネットワーク
4	マルチドメイン マネージャまたはその他
5	クラウド

◦ String accountType：アカウントのタイプ。"11" のように引用符で囲まれた値を渡す必要があります。

アカウント タイプの使用可能な値は次のとおりです。

整数	アカウントの種類
2	VMware
[6]	HYPERV
9	REDHAT_KVM
10	XENDESKTOP
11	UCSM
12	NETAPP
18	NETAPP_DFM
15	HP
16	CISCO_CIMC
17	EMC_VNX

整数	アカウントの種類
18	IPMI
19	LOAD_BALANCERS
20	EMC_VMAX
24	EMC_VPLEX
22	WHIPTAIL
23	EMC_ISILON
25	EMC_NEW_VNX
26	EMC_NEW_VNX_BLOCK
27	EMC_NEW_VNX_UNIFIED
36	HP_OA
29	CAT_LDAP
30	CAT_LDAP_CLEANUP
31	VCE_VISION_IO
32	EMC_RECOVERPOINT
33	UCS_INVICTA_APPLIANCE
34	UCS_INVICTA_SCALING
35	EMC_NEW_VNX_BLOCK_HTTP
36	EMC_VNXE

- String deviceCategory：コンピューティング、ストレージ、ネットワークなどのデバイスのカテゴリ。
- String description：オプション。このアカウントの説明。
- String contact：オプション。管理者またはアカウント責任者の連絡先に使用できる電子メール アドレスです。
- String destinationIPAddress：このアカウントの宛先 IP アドレス。引用符で囲まれた IP アドレスを渡す必要があります。

- String login：このアカウントがエレメント マネージャにアクセスするために使用するログイン ID。たとえば、Cisco UCS Manager のアカウントはこのログイン ID を使用して Cisco UCS Manager にアクセスします。このユーザ名は Cisco UCS Manager の有効なアカウントである必要があります。
- String password：ログイン ID に関連付けられるパスワード。
- String enablepassword：オプション。このアカウントにパスワードをイネーブルにするには値を true として渡します。
- String protocol：アカウントとの通信に使用するプロトコル。有効な値は Telnet および SSH です。
- int port：エレメント マネージャにアクセスするために使用するポート。
- infraAccountSupportDetailsInfo：インフラ アカウントの詳細。
  infraAccountSupportDetailsInfo には、次のパラメータが含まれます。
  - String spAIpAddress：オプション。VNX デバイスのストレージ プロセッサ A の IP アドレス。
  - String spBIpAddress：オプション。VNX デバイスのストレージ プロセッサ B の IP アドレス。
  - String blockAccessUserName：オプション。VNX デバイスのブロック アクセスのユーザ名。
  - String blockAccessPwd：オプション。VNX デバイスのブロック アクセスのパスワード。
  - String sshIpAddress：オプション。SSH サーバの IP アドレス。引用符で囲まれた IP アドレスを渡す必要があります。
  - String sshUsername：オプション。このアカウントが SSH サーバにアクセスするために使用するユーザ名。
  - String sshPassword：オプション。SSH ユーザ名に関連付けられたパスワード。
  - int sshPort：オプション。SSH サーバへのアクセスに使用されるポート。
  - String domain：オプション。アカウントに関連付けられたドメイン。
  - String serviceProvider：オプション。このアカウントに関連付けられたサービス プロバイダーの名前（ある場合）。

コード

```
public class UserAPICreateInfraAccountExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.187",
"6C6416AD90704DF495A6B4D0A75A0BB1", "https", 443);
  UserAPIAccounts instance = new UserAPIAccounts(server);
  InfraAccountDetails infraAccountDetails = new InfraAccountDetails();
  infraAccountDetails.setAccountName("AccName");
  infraAccountDetails.setPodName("PodName");
  infraAccountDetails.setAccountCategory(2);
  infraAccountDetails.setAccountType("11");
  infraAccountDetails.setProtocol("http");
  infraAccountDetails.setPort(80);
  infraAccountDetails.setDestinationIPAddress("172.29.32.14");
  infraAccountDetails.setLogin("admin");
  infraAccountDetails.setPassword("admin");
  boolean isCreated = instance.userAPICreateInfraAccount(infraAccountDetails);
  System.out.println("is Account Created ?"+isCreated);
 }
}
```

結果

物理アカウントが作成されます。戻り値は作成が正常に行われた場合は true です。

実装

アカウント情報を含む InfraAccountDetails オブジェクトのインスタンスを作成してから、userAPICreateInfraAccount を呼び出して物理アカウントを作成します。

関連項目

アカウントのリスト化

物理アカウントの削除

アカウントのリスト化

目標

Cisco UCS Director の物理アカウントおよび仮想アカウントを取得します。

コンテキスト

Cisco UCS Director 内の既存のリソースまたはアカウントを特定します。

前提条件

ユーザには、Cisco UCS Director 内のすべてのアカウントを表示する権限が必要です。

REST の URL

```
/app/api/rest?formatType=json&opName=accounts:userAPIGetAllPhysicalInfraAccounts&opData={}
```

コンポーネント

なし

コード

```
public class UserAPIGetAllAccountsExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIAccounts instance = new UserAPIAccounts(server);
  List<String> accountNameList = instance.userAPIGetAllAccounts();
  for (int i = 0; i < accountNameList.size(); ++i)
   {
   System.out.println("Name "+accountNameList.get(i));
   }
  }
}
```

結果

物理アカウントおよび仮想アカウントのリストが表示されます。

実装

既存の物理アカウントおよび仮想アカウントの詳細を取得するには、userAPIGetAllAccounts API を呼び出します。

関連項目

物理アカウントの作成

物理アカウントの削除

物理アカウントの削除

目標

Cisco UCS Director から物理アカウントを削除します。

コンテキスト

物理アカウントは、アカウントがマッピングされているグループに属しているユーザが削除できます。

前提条件

物理アカウントが使用可能である必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=accounts:userAPIDeleteInfraAccount&opData={param0:"UCSM-150"}
```

コンポーネント

String Account name：削除する必要がある物理アカウントの名前。

コード

```
public class UserAPIDeleteInfraAccountExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIAccounts instance = new UserAPIAccounts(server);
  boolean isDeleted = instance.userAPIDeleteInfraAccount("UCSAccount");
  System.out.println("Is the Account deleted ? :" + isDeleted);
 }
}
```

結果

物理アカウントが正常に削除された場合、結果は true となります。

実装

既存の物理アカウントを削除するには、物理アカウント名を渡して userAPIDeleteInfraAccount API を呼び出します。

関連項目

物理アカウントの作成

アカウントのリスト化

仮想データセンターの管理

VDC の作成

目標

VM がプロビジョニングされる仮想データセンター（VDC）を作成します。

コンテキスト

VM をプロビジョニングする VDC を作成します。

前提条件

クラウドとグループが存在する必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPICreateVDC&opData={param0:{"vdcName":"Vdc"
,"vdcDescription":"VDC","cloudName":"VMware-Cloud","groupName":2,"approver1":"","approver2":
"","vdcSupportEmail":"test@cisco.com","vdcCustomerNoticationEmail":"","systemPolicy":
"VmwareCloudSysPolicy","deploymentPolicy":"","slaPolicy":"","computingPolicy":"","storagePolicy":
"","networkPolicy":"","costModel":"","isLocked":false,"isDeletable":false,
"inactivityPeriodForDeletion":1000}}
```

コンポーネント

APIVDCDetails

コード

```
public class CreateVDCExample
 {
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APIVDCDetails vdcDetails = new APIVDCDetails();
  vdcDetails.setVdcName("Vdc");
  vdcDetails.setVdcDescription("VDC");
  vdcDetails.setCloudName("VMware-Cloud");
  vdcDetails.setGroupName(2);
  vdcDetails.setApprover1("");
  vdcDetails.setApprover2("");
  vdcDetails.setVdcSupportEmail("test@cisco.com");
  vdcDetails.setVdcCustomerNoticationEmail("");
  //System policy is optional
  vdcDetails.setSystemPolicy("VmwareCloudSysPolicy");
  //System policy is optional
  vdcDetails.setSlaPolicy("");
  //System policy is optional
  vdcDetails.setComputingPolicy("");
  //netowrk policy is optional
  vdcDetails.setNetworkPolicy("");
  //storage  policy is optional
  vdcDetails.setStoragePolicy("");
  //Cost model is optional
  vdcDetails.setCostModel("");
  //vdc locked  is optional
  vdcDetails.setLocked(false);
  //Deletable is optional
  vdcDetails.setDeletable(false);
  vdcDetails.setInactivityPeriodForDeletion(1000);
  boolean isVDCCreated = instance.userAPICreateVDC(vdcDetails);
  System.out.println("is VDC Created "+isVDCCreated);
 }
}
```

結果

VDC が正常に作成された場合、結果は true となります。

実装

必要な情報を渡すことで APIVDCDetails を作成し、userAPICreateVDC を呼び出して VDC を作成します。

関連項目

VDC のリスト化

VDC のエクスポート

VDC のインポート

VDC リソース制限の取得, （41 ページ）

コスト モデルの取得, （43 ページ）

VDC の削除

VDC のリスト化

目標

ユーザ グループ内の VDC のリストを取得します。

コンテキスト

VM をプロビジョニングするときに必要な VDC を選択するために、グループ内の VDC のリストを取得します。

前提条件

ログイン ユーザがグループに割り当てられている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIGetAllVDCs&opData={}
```

コンポーネント

なし

コード

```
public class UserAPIGetAllVDCExample
{
 public static void main(String[] args) throws Exception
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APITabularReport tabularReport = instance.userAPIGetAllVDCs();
  System.out.println("No. Of VDC :"+tabularReport.getRowCount());
  System.out.println("No. Of Column : "+tabularReport.getColumnMetaData());
  }
}
```

結果

ユーザ グループ内の VDC のリストが返されます。

実装

ユーザ グループ内のすべての VDC を取得するには、userAPIGetAllVDCs API を呼び出します。

関連項目

VDC の作成

VDC のエクスポート

VDC のインポート

VDC リソース制限の取得, （41 ページ）

コスト モデルの取得, （43 ページ）

VDC の削除

VDC のエクスポート

目標

Cisco UCS Director サーバからローカル システムに VDC をエクスポートします。

コンテキスト

別の Cisco UCS Director サーバに同じ VDC のセットを作成します。

前提条件

エクスポートする VDC の名前。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIExportVDC&opData={param0:"vDCIT"}
```

コンポーネント

vdcName：エクスポートする VDC の名前。

コード

```
public class ExportVDCExample
 {
 public static void main(String[] args) throws Exception
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  String exportVDCString = instance.userAPIExportVDC("sample_VDC");
  System.out.println("Export VDC as "+exportVDCString);
 }
}
```

結果

sample_VDC VDC がローカル システムにエクスポートされます。

実装

ローカル システムに VDC をエクスポートするには、userAPIExportVDC API を使用して VDC 名を渡します。

関連項目

VDC のリスト化

VDC の作成

VDC のインポート

VDC リソース制限の取得, （41 ページ）

コスト モデルの取得, （43 ページ）

VDC の削除

VDC のインポート

目標

ローカル システムから Cisco UCS Director に VDC をインポートします。

コンテキスト

Cisco UCS Director に外部 VDC をインポートします。

前提条件

インポートする必要がある VDC がローカル システムで使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIImportVDC&opData={param0:"importvdcasstring"}
```

コンポーネント

vdcName：インポートする VDC の名前。

コード

```
public class ImportVDCExample
{
 public static void main(String[] args) throws Exception
 {
 CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  //You have to pass the exported VDC String as importVDC parameter
  VDC vdcObject = instance.userAPIImportVDC("exportVDCAsString" );
  System.out.println("VDC Name = "+vdcObject.getVdcName());
 }
}
```

結果

インポートされた VDC が Cisco UCS Director に表示されます。

実装

Cisco UCS Director に VDC をインポートするには、userAPIImportVDC API を呼び出して VDC 名を渡します。

関連項目

VDC の作成

VDC のリスト化

VDC のエクスポート

VDC リソース制限の取得, （41 ページ）

コスト モデルの取得, （43 ページ）

VDC の削除

VDC リソース制限の取得

目標

VDC のリソース制限を取得します。

コンテキスト

VDC に使用されているリソース制限を特定します。

前提条件

コンテナまたは VDC が定義されている必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIGetVDCResourceLimits&opData=
{param0:"bmtest"}
```

応答

```
{ "serviceResult":{"provisionedNoOfvCPUsLimit":0,"provisionedMemGBLimit":0.0,
"provisionedDiskGBLimit":0.0,"halfWidthPhysicalServerLimit":0,"fullWidthPhysicalServerLimit":0},

 "serviceError":null, "serviceName":"InfraMgr", "opName":"fenced:userAPIGetVDCResourceLimits"
 }
```

コンポーネント

なし

コード

```
public class GetVDCResourceLimits
{
 public static void main(String[] args) throws Exception
 {
  CuicServer api = CuicServer.getAPI("172.29.110.241",
"EDDF69C4D9AA4E4FA8F14EFD57B90402", "http", 80);
  UserAPIFencedContainer fenced = new UserAPIFencedContainer(api);
  APIResourceLimitParams params = new APIResourceLimitParams();
  params.setVdcName("bmtest");
  APIResourceLimitResponse response = fenced.userAPIGetVDCResourceLimits(params);
  System.out.println( " Response is \n Full Width
Limits"+response.getFullWidthPhysicalServerLimit());
  System.out.println( " Half Width Limits"+response.getHalfWidthPhysicalServerLimit());

  System.out.println( " Provisioned Disk GB "+response.getProvisionedDiskGBLimit());
  System.out.println( " Provisioned Memory"+response.getProvisionedMemGBLimit());
  System.out.println( " Provisioned vCPU Limits
"+response.getProvisionedNoOfvCPUsLimit());
 }
}
```

結果

VDC リソースに対する制限について、フル幅制限、ハーフ幅制限、プロビジョニング済みディスク、プロビジョニングされたメモリ、およびプロビジョニングされた vCPU の制限が表示されます。

実装

userAPIGetVDCResourceLimits API を使用して、VDC 名を渡し、リソース制限の設定を表示します。

関連項目

VDC の作成, （36 ページ）

VDC のリスト化

VDC のエクスポート

VDC のインポート

コスト モデルの取得, （43 ページ）

VDC の削除

コスト モデルの取得

目標

VDC で利用可能なリソースのコスト モデルを取得します。

コンテキスト

指定したリソース（CPU、メモリ、ディスクなど）の VM と物理サーバについて、時間ごとと月ごとのコスト モデルを取得します。

前提条件

VDC およびコスト モデルが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=chargeback:userAPIGetCostModel&opData=
{param0:{"vdcName":"CostModel_Vdc","costModelResources":[{"name":"CPU","value":"1"},
{"name":"Memory","value":"1"},{"name":"Disk","value":"1"},{"name":"NetworkRx",
"value":"1"},{"name":"NetworkTx","value":"1"},{"name":"BMCPU","value":"1"},
{"name":"BMMemory","value":"1"},{"name":"BMDisk","value":"1"},
{"name":"BladeType","value":"Half"}]}}
```

応答

```
{ "serviceResult":{"costModelRequestedInfo":{"vDC Name":"CostModel_Vdc",
"Memory (GB)":"1","Disk (GB)":"1","NetworkRx (GB)":"1","NetworkTx (GB)":"1",
"CPU (GHz)":"1","BMMemory (GB)":"1","BMDisk (GB)":"1","Blade Type":"Half",
"BMCPU (Cores)":"1"},"vmCostModel":{"oneTimeItemCost":100.0,"monthlyCost":2180.0,
"cpuCostModel":{"cpuGhzCostPerHour":1.0,"cpuCostPerCore":0.0,"totalCPUCost":720.0},
"diskCostModel":{"diskGBCostPerHour":1.0,"totalDiskCost":720.0},"memoryCostModel":
{"memoryGBCostPerHour":1.0,"totalMemoryCost":720.0},"networkCostModel":
{"netRxCostPerGB":10.0,"netTxCostPerGB":10.0,"totalNetworkCost":20.0}},
"bMCostModel":{"oneTimeItemCost":100.0,"monthlyCost":2880.0,"cpuCostModel":
{"cpuGhzCostPerHour":0.0,"cpuCostPerCore":1.0,"totalCPUCost":720.0},
"diskCostModel":{"diskGBCostPerHour":1.0,"totalDiskCost":720.0},"memoryCostModel":
{"memoryGBCostPerHour":1.0,"totalMemoryCost":720.0},"bladeCostModel":
{"fullBladeCostPerHour":0.0,"halfBladeCostPerHour":1.0,"totalBladeCost":720.0}}},
"serviceError":null, "serviceName":"InfraMgr", "opName":"chargeback:userAPIGetCostModel"
 }
```

コンポーネント

- VdcName：VDC の名前。

可能な名前：値のペアは次のとおりです。

- CPU：オプション。プロビジョニングされたCPUの上限（GH□ 単位またはコア単位）。
- Memory：オプション。プロビジョニングされたメモリの上限（GB 単位）。
- NetworkRx：オプション。データ トラフィックの受信レート。
- NetworkTx：オプション。データ トラフィックの送信レート。
- Disk：オプション。プロビジョニングされたディスクの上限（GB 単位）。
- BMCPU：オプション。プロビジョニングされた BM の CPU の上限（GH□ 単位またはコア単位）。
- BMMemory：オプション。プロビジョニングされた BM メモリの上限（GB 単位）。
- BMDisk：オプション。プロビジョニングされた BM のディスクの上限（GB 単位）。
- BladeType：オプション。ブレードのタイプ。

コード

```
public class GetCostModel
{
 public static void main(String[] args)throws Exception
{
  CuicServer server = CuicServer.getAPI("<IP address>",
            "<REST Key>", "https", 443);
  UserAPIChargeBack instance = new UserAPIChargeBack(api);
  List<APINameValue> requestParam = new ArrayList<APINameValue>();
  APIGetCostModelParams costParams = new APIGetCostModelParams();
  costParams.setVdcName("CostModel_Vdc");
  APINameValue value=new APINameValue();

  value.setName("CPU");
  value.setValue("1");
  requestParam.add(value);
  value=new APINameValue();
  value.setName("Memory");
  value.setValue("1");
  requestParam.add(value);
  value=new APINameValue();
  value.setName("Disk");
  value.setValue("1");
  requestParam.add(value);
  value=new APINameValue();
  value.setName("NetworkRx");
  value.setValue("1");
  requestParam.add(value);
  value=new APINameValue();
  value.setName("NetworkTx");
  value.setValue("1");
  value=new APINameValue();
  value.setName("BMCPU");
  value.setValue("1");
  requestParam.add(value);
   costParams.setCostModelResources(requestParam);
   APIGetCostModelResponse response = instance.userAPIGetCostModel(costParams);
   APIVMCostModel vmCostModel = response.getVmCostModel();
   APIPhysicalServerCostModel bmCostModel = response.getBMCostModel();
   if(vmCostModel != null){
    System.out.println(vmCostModel.getOneTimeItemCost());
   APICPUCostModel cpuModel =  vmCostModel.getCpuCostModel();
    System.out.println(cpuModel.getTotalCPUCost());
   }
  if(bmCostModel != null){
   System.out.println(bmCostModel.getOneTimeItemCost());
   APICPUCostModel bmCPUModel = bmCostModel.getCpuCostModel();
         System.out.println(bmCPUModel.getTotalCPUCost());
  }
 }
 }
```

結果

VM 内および物理サーバ内のリソースのコスト モデルが表示されます。

実装

userAPIGetCostModel API を呼び出し、コスト モデルを表示させるリソースとともに VDC 名を渡します。

関連項目

VDC の作成, （36 ページ）
VDC のリスト化, （38 ページ）
VDC のエクスポート, （39 ページ）
VDC のインポート, （40 ページ）
VDC リソース制限の取得, （41 ページ）
VDC の削除, （46 ページ）

VDC の削除

目標

Cisco UCS Director から VDC を削除します。

コンテキスト

Cisco UCS Director から使用されていない VDC を削除します。

前提条件

ログイン ユーザに VDC を削除する権限が必要です。

REST の URL

```
Not Applicable
```

コンポーネント

vdcName：削除する VDC の名前。

コード

```
public class DeleteVDCExample
{
 public static void main(String[] args) throws Exception
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  DeleteVDCConfig instance = new DeleteVDCConfig(server);
  instance.setVdcName("");
  instance.setRollbackRequest(false);
  boolean isDeleted = instance.execute();
  System.out.println(" is VDC Deleted "+isDeleted);
 }
}
```

結果

VDC が正常に削除された場合、結果は true となります。

実装

VDC 名を渡すことで、VDC を削除します。

関連項目

VDC の作成

VDC のリスト化

VDC のエクスポート

VDC のインポート

VDC リソース制限の取得, （41 ページ）

コスト モデルの取得, （43 ページ）

仮想インフラストラクチャ ポリシーの管理

仮想インフラストラクチャ ポリシーの作成

目標

Fenced 仮想アプリケーション コンテナの仮想インフラストラクチャ ポリシーを作成します。仮想インフラストラクチャ ポリシーは、使用する VM やプロビジョニングするコンテナのタイプを定義します。また、このポリシーは、この特定のアカウントに関連付ける PNSC アカウントも定義します。

コンテキスト

仮想インフラストラクチャ ポリシーは、アプリケーション コンテナ テンプレートの作成に使用されます。このテンプレートでは、さまざまなネットワーク（DFA ネットワークを含む）で使用するアプリケーション コンテナを作成できます。

前提条件

VMware バーチャル アカウントが存在する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=fenced:
userAPICreateServiceContainerVirtualInfraPolicy&opData={param0:{"policyName":
"vipFenc","policyDescription":"","virtualAccountName":"vc117",
"isGatewayRequired":false,"gatewayPolicyName":"","isF5LoadBalancerRequired":
false,"f5LoadBalancerPolicyName":""}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"fenced:userAPICreateServiceContainerVirtualInfraPolicy" }
```

コンポーネント

- policyName：仮想インフラストラクチャ ポリシーの名前。
- policyDescription：オプション。仮想インフラストラクチャ ポリシーの説明。
- virtualAccountName：仮想インフラストラクチャ ポリシーを定義する仮想アカウントの名前。
- isGatewayRequired：Fenced コンテナにゲートウェイが必要な場合、このパラメータをTrueに設定します。
- gatewayPolicyName：isGatewayRequired パラメータが True に設定された場合に、ゲートウェイのポリシー名を指定します。
- isF5LoadBalancerRequired：F5 ロードバランサ サービスが必要とされる場合、このパラメータを True に設定します。
- f5LoadBalancerPolicyName：isF5LoadBalancerRequired パラメータが True に設定された場合に、F5 ロードバランサのポリシー名を指定します。

コード

```
public class FencedContainerVirtualInfraPolicyCreateExample
{
 public static void main(String[] args) throws Exception
{
   CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
   UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
   FencedContainerVirtualInfrastructurePolicy policy = new
FencedContainerVirtualInfrastructurePolicy();
   policy.setPolicyName("VIPolicy");
   policy.setPolicyDescription("VIPolicy1 Description");
   policy.setVirtualAccountName("vm117");
   policy.setGatewayRequired(false);
   policy.setF5LoadBalancerRequired(false);
   boolean obj = instance.userAPICreateServiceContainerVirtualInfraPolicy( policy );
   System.out.println("Created Sucessfuly : "+obj);
 }
}
```

結果

仮想インフラストラクチャ ポリシーが正常に追加された後に、True を返します。

実装

userAPICreateServiceContainerVirtualInfraPolicy API を呼び出し、必要な情報とともに仮想アカウント名を渡します。

関連項目

仮想インフラストラクチャ ポリシーの取得
仮想インフラストラクチャ ポリシーの変更
仮想インフラストラクチャ ポリシーの削除

仮想インフラストラクチャ ポリシーの取得

目標

ポリシー名を渡すことで、仮想インフラストラクチャ ポリシーの詳細情報を取得します。

コンテキスト

設定されたポリシーの詳細を表示します。

前提条件

仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=fenced:
userAPIGetServiceContainerVirtualInfraPolicy&opData={param0:"vipPolicy"}
```

応答


```
{ "serviceResult":[{"policyId":2,"policyName":" vipPolicy ",
"policyDescription":"","containerType":"Fenced Virtual","account":"V100",
"vnmcConfig":{"vnmcAccount":null,"vnmcFirewallPolicies":null},"apicConfig":
{"applicationProfile":null},"gatewayConfig":{"gatewayRequired":false,
"gatewayPolicy":"","summaryArea":null,"gatewayType":"No Gateway"},"f5LBConfig":
{"f5LoadBalancerRequired":false,"f5LoadBalancerPolicy":"","summaryArea":null,
"f5LoadBalancerType":"No F5 Load Balancer"},"dfaConfig":{"vsgEnabled":false,
"accountId":"","switchType":"","switchName":"","alternateSwitchName":"",
"serviceNetworkConfiguration":null,"l3Network":false,"fabricOrganization":"",
"fabricPartition":"","fabricNetwork":null,"serviceNetworkMobilityDomain":null,
"autoServiceNetworkMobilityDomain":true,"hostNetworkConfiguration":null,
"mobilityDomain":null,"autoSelectMobilityDomain":true,"partitionParamsLabel":null,
"dciID":null,"extendPrtnAcrossFabric":true,"serviceNodeIpAddress":null,
"dnsServer":null,"secondaryDNSServer":null,"multiCastGroupAddress":null,
"profileName":"None","profileParamsLabel":null,"includeBorderLeafRt":null},
"fabricASAConfig":{"ipSubnetPoolPolicy":null,"internalConfigurationLabel":null,
"internalNetworkName":null,"internalNetworkRole":null,
"internalNetworkGatewayIP":null,"internalNetworkMask":null,
"internalNetworkProfile":null,"internalNetworkMobilityDomain":null,
"autoSelectInternalNetworkMobilityDomain":true,"externalConfigurationLabel":null,
"externalPartitionProfile":null,"externalNetworkName":null,"externalNetworkRole":null,
"externalNetworkGatewayIP":null,"externalNetworkMask":null,
"externalNetworkProfile":null,"externalNetworkMobilityDomain":null,
"autoSelectExternalNetworkMobilityDomain":true},"fabricASAvConfig":
{"ipSubnetPoolPolicy":null,"internalConfigurationLabel":null,"internalNetworkName":null,
"internalNetworkRole":null,"internalNetworkGatewayIP":null,"internalNetworkMask":null,
"internalNetworkProfile":null,"internalSwitchType":"","internalSwitchName":
"","internalNetworkMobilityDomain":null,"autoSelectInternalNetworkMobilityDomain":true,
"externalConfigurationLabel":null,"externalPartitionProfile":null,"externalNetworkName":null,
"externalNetworkRole":null,"externalNetworkGatewayIP":null,"externalNetworkMask":null,
"externalNetworkProfile":null,"externalSwitchType":"","externalSwitchName":
"","externalNetworkMobilityDomain":null,"autoSelectExternalNetworkMobilityDomain":true},
"fabricF5Config":{"serviceConfigurationLabel":null,"serviceNetworkName":null,
"serviceNetworkRole":null,"serviceNetworkGatewayIP":null,"serviceNetworkMask":null,
"serviceNetworkProfile":null,"f5ServiceNetworkMobilityDomain":null,
"autoSelectServiceNetworkMobilityDomain":true}}], "serviceError":null,
 "serviceName":"InfraMgr", "opName":"fenced:userAPIGetServiceContainerVirtualInfraPolicy"
 }
```


コンポーネント

policyName：仮想インフラストラクチャ ポリシーの名前。

コード

```
public class retrieveAPI
{
public static void main(String[] args) throws Exception
{
 CuicServer server = CuicServer.getAPI("10.23.210.118","ADFGLKJSHFJ23478234HJBFJGH",
"https", 443);
 UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
 List obj = instance.userAPIGetServiceContainerVirtualInfraPolicy( "x" );
 System.out.println(obj);
 }
}
```

結果

ポリシーの詳細が正常に取得された場合、結果は true となります。

実装

userAPIGetServiceContainerVirtualInfraPolicy API を呼び出し、仮想インフラストラクチャ ポリシー名を渡します。

関連項目

仮想インフラストラクチャ ポリシーの作成

仮想インフラストラクチャ ポリシーの変更

仮想インフラストラクチャ ポリシーの削除

仮想インフラストラクチャ ポリシーの変更

目標

指定された詳細の内容で Fenced コンテナの仮想インフラストラクチャ ポリシーを更新します。ポリシー名は編集できません。

コンテキスト

コンテナに定義された仮想インフラストラクチャ ポリシーを更新します。更新されたポリシーは、新しいアプリケーション コンテナ テンプレートの作成に使用されます。

前提条件

仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=fenced:
userAPIUpdateServiceContainerVirtualInfraPolicy&opData={param0:
{"policyName":"Testing","policyDescription":"Testing Update Description ",
"virtualAccountName":"VNX_Cloud169","isGatewayRequired":false,
"gatewayPolicyName":"","isF5LoadBalancerRequired":false,"f5LoadBalancerPolicyName":""}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"fenced:userAPIUpdateServiceContainerVirtualInfraPolicy" }
```

コンポーネント

- policyName：仮想インフラストラクチャ ポリシーの名前。
- policyDescription：オプション。仮想インフラストラクチャ ポリシーの説明。
- virtualAccountName：仮想インフラストラクチャ ポリシーを定義する仮想アカウントの名前。
- isGatewayRequired：Fenced コンテナにゲートウェイが必要な場合、このパラメータをTrueに設定します。
- gatewayPolicyName：isGatewayRequired パラメータが True に設定された場合に、ゲートウェイのポリシー名を指定します。
- isF5LoadBalancerRequired：F5 ロードバランサ サービスが必要とされる場合、このパラメータを True に設定します。
- f5LoadBalancerPolicyName：isF5LoadBalancerRequired パラメータが True に設定された場合に、F5 ロードバランサのポリシー名を指定します。

コード

```
public class FencedContainerVirtualInfraPolicyUpdateExample
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
   UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
   FencedContainerVirtualInfrastructurePolicy policy = new
FencedContainerVirtualInfrastructurePolicy();
   policy.setPolicyName("VIPolicy");
   policy.setPolicyDescription("VIPolicy1 Description modified");
   policy.setVirtualAccountName("vm117");
   policy.setGatewayRequired(false);
   policy.setF5LoadBalancerRequired(false);
   boolean obj = instance.userAPIUpdateServiceContainerVirtualInfraPolicy( policy );
   System.out.println("Created Sucessfuly : "+obj);
 }
}
```

結果

仮想インフラストラクチャ ポリシーが正常に更新された場合、結果は True となります。

実装

userAPIUpdateServiceContainerVirtualInfraPolicy API を呼び出し、更新の必要な詳細内容とともに仮想インフラストラクチャ ポリシー名と仮想アカウント名を渡します。

関連項目

仮想インフラストラクチャ ポリシーの作成

仮想インフラストラクチャ ポリシーの取得

仮想インフラストラクチャ ポリシーの削除

仮想インフラストラクチャ ポリシーの削除

目標

Cisco UCS Director から仮想インフラストラクチャ ポリシーを削除します。

コンテキスト

グループに属しているユーザは、仮想インフラストラクチャ ポリシーを削除できます。

前提条件

仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=fenced:
userAPIDeleteServiceContainerVirtualInfraPolicy
&opData={param0:{"policyName":"vipFenc"}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"fenced:userAPIDeleteServiceContainerVirtualInfraPolicy" }
```

コンポーネント

- policyName：仮想インフラストラクチャ ポリシーの名前。

コード

```
public class DeleteServiceContainerVirtualInfraPolicy
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
 UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
 FencedContainerVirtualInfrastructurePolicy policy = new
FencedContainerVirtualInfrastructurePolicy();
 policy.setPolicyName("VIPolicy");
 boolean obj = instance.userAPIDeleteServiceContainerVirtualInfraPolicy( policy );
 System.out.println(obj);
 }
}
```

結果

仮想インフラストラクチャ ポリシーが正常に削除された場合、結果は True となります。

実装

userAPIDeleteServiceContainerVirtualInfraPolicy API を呼び出し、仮想インフラストラクチャ ポリシー名を渡します。

関連項目

仮想インフラストラクチャ ポリシーの作成

仮想インフラストラクチャ ポリシーの取得

仮想インフラストラクチャ ポリシーの変更

APIC 仮想インフラストラクチャ ポリシーの管理

APIC 仮想インフラストラクチャ ポリシーの作成

目標

APIC コンテナの仮想インフラストラクチャ ポリシーを作成します。

コンテキスト

仮想インフラストラクチャ ポリシーは、サービス コンテナ テンプレートの作成に使用されます。

前提条件

バーチャル アカウントが存在する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
apic:userAPICreateServiceContainerVirtualInfraPolicy&opData={param0:
{"policyName":"VIPolicy1","policyDescription":"Create VIP Policy","containerType":
"APIC","applicationProfileName":" appprofile"}}
```

API の実行成功時の応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPICreateServiceContainerVirtualInfraPolicy" }
```

API の実行失敗時の応答

REST URL に指定されたアプリケーション プロファイルが存在しない場合、API は次の例外をスローします。この例外では、有効なアプリケーション プロファイル名を使用するか、userAPICreateApplicationProfile API を使用して新しいアプリケーション プロファイルを作成するように提案されます。

```
{ "serviceResponse":null, "serviceError":"REMOTE_SERVICE_EXCEPTION:
Application profile test does not exist, Please use userAPICreateApplicationProfile
api to create new application profile.", "serviceName":"InfraMgr",
"opName":"apic:userAPICreateServiceContainerVirtualInfraPolicy" }
```

コンポーネント

- policyName：仮想インフラストラクチャ ポリシーの名前。
- containerType：コンテナのタイプは APIC であることが必要です。
- applicationProfileName：コンテナの作成に使用するアプリケーション プロファイルの名前。

コード

```
public class APICContainerVirtualInfraPolicyCreateExample
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
  UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
  APICContainerVirtualInfraStructurePolicy policy = new
APICContainerVirtualInfraStructurePolicy();
  policy.setPolicyName("VIPolicy1");
  policy.setPolicyDescription("VIPolicy1 Description");
  policy.setApplicationProfileName("applicationProfileName");
  boolean obj = instance.userAPICreateServiceContainerVirtualInfraPolicy(policy);
  System.out.println("Created Sucessfuly : " + obj);

  /**
   * Update the service Container Virtual Infra Policy
   */

  obj = instance.userAPIUpdateServiceContainerVirtualInfraPolicy(policy);
  System.out.println(obj); }
}
```

結果

仮想インフラストラクチャ ポリシーが正常に追加された場合、結果は true となります。

実装

userAPICreateServiceContainerVirtualInfraPolicy API を呼び出し、必要な情報とともにアプリケーション プロファイル名を渡します。

関連項目

すべての APIC 仮想インフラストラクチャ ポリシーのリスト化

APIC 仮想インフラストラクチャ ポリシーの取得

APIC 仮想インフラストラクチャ ポリシーの変更

APIC 仮想インフラストラクチャ ポリシーの削除

すべての APIC 仮想インフラストラクチャ ポリシーのリスト化

目標

Cisco UCS Director 内のすべての APIC 仮想インフラストラクチャ ポリシーを取得します。

コンテキスト

Cisco UCS Director 内の APIC 仮想インフラストラクチャ ポリシーのリストを取得します。

前提条件

要求ユーザは、APIC 仮想インフラストラクチャ ポリシーのリストを取得する権限を持っている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=
apic:userAPIGetAllServiceContainerVirtualInfraPolicies&opData={}
```

コンポーネント

なし

コード

```
public class GetAllAPICContainerVirtualInfraPolicies
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
 UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
  List<APICContainerVirtualInfraStructurePolicy> list =
instance.userAPIGetAllServiceContainerVirtualInfraPolicies();
  for (Iterator iterator = list.iterator(); iterator.hasNext();) {
  APICContainerVirtualInfraStructurePolicy apicContainerVirtualInfraStructurePolicy =
 (APICContainerVirtualInfraStructurePolicy) iterator
   .next();
   System.out.println(" Profile Name :
"+apicContainerVirtualInfraStructurePolicy.getPolicyName());
   System.out.println(" Profile Description :
"+apicContainerVirtualInfraStructurePolicy.getPolicyDescription());
   System.out.println(" Application Profile Name :
"+apicContainerVirtualInfraStructurePolicy.getApplicationProfileName());
  }
}
```

結果

API によって、Cisco UCS Director 内の APIC 仮想インフラストラクチャ ポリシーのリストが返されます。

実装

実装は必要ありません。

関連項目

APIC 仮想インフラストラクチャ ポリシーの作成

APIC 仮想インフラストラクチャ ポリシーの取得

APIC 仮想インフラストラクチャ ポリシーの変更

APIC 仮想インフラストラクチャ ポリシーの削除

APIC 仮想インフラストラクチャ ポリシーの取得

目標

ポリシー名を渡すことで、仮想インフラストラクチャ ポリシーの詳細情報を取得します。

コンテキスト

設定されたポリシーの詳細を表示します。

前提条件

仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=
apic:userAPIGetServiceContainerVirtualInfraPolicy&opData={param0:"testPolicy"}
```

コンポーネント

- policyName：削除する仮想インフラストラクチャ ポリシーの名前。

コード

```
public class GetAPICContainerVirtualInfraPolicyExample
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
 UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
  List<APICContainerVirtualInfraStructurePolicy> list =
instance.userAPIGetServiceContainerVirtualInfraPolicy("apicPolicy");
  for (Iterator iterator = list.iterator(); iterator.hasNext();) {
   APICContainerVirtualInfraStructurePolicy apicContainerVirtualInfraStructurePolicy
= (APICContainerVirtualInfraStructurePolicy) iterator
      .next();
   System.out.println(" Profile Name :
"+apicContainerVirtualInfraStructurePolicy.getPolicyName());
   System.out.println(" Profile Description :
"+apicContainerVirtualInfraStructurePolicy.getPolicyDescription());
   System.out.println(" Application Profile Name :
"+apicContainerVirtualInfraStructurePolicy.getApplicationProfileName());
   }
}
```

結果

ポリシーの詳細が正常に取得された場合、結果は true となります。

実装

userAPIGetServiceContainerVirtualInfraPolicy API を呼び出し、仮想インフラストラクチャ ポリシー名を渡します。

関連項目

APIC 仮想インフラストラクチャ ポリシーの作成

すべての APIC 仮想インフラストラクチャ ポリシーのリスト化

APIC 仮想インフラストラクチャ ポリシーの変更

APIC 仮想インフラストラクチャ ポリシーの削除

APIC 仮想インフラストラクチャ ポリシーの変更

目標

指定された詳細の内容で APIC コンテナの仮想インフラストラクチャ ポリシーを更新します。ポリシー名は編集できません。

コンテキスト

APIC コンテナに定義された仮想インフラストラクチャ ポリシーを更新します。更新されたポリシーは、新しいサービス コンテナ テンプレートの作成に使用されます。

前提条件

APIC 仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=apic:
userAPIUpdateServiceContainerVirtualInfraPolicy&opData={param0"policyName":
"testPolicy","policyDescription":"updated testPolicy ","containerType":"APIC",
"applicationProfileName":"appprofile"}}
```

API の実行成功時の応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPIUpdateServiceContainerVirtualInfraPolicy" }
```

API の実行失敗時の応答

REST URL に指定されたアプリケーション プロファイルが存在しない場合、API は次の例外をスローし、有効なアプリケーション プロファイル名を使用するよう提案します。

```
{ "serviceResponse":null, "serviceError":"REMOTE_SERVICE_EXCEPTION:
Application profile ghfgh does not exist, Please provide available
 application profile name", "serviceName":"InfraMgr",
"opName":"apic:userAPIUpdateServiceContainerVirtualInfraPolicy" }
```

コンポーネント

- policyName：仮想インフラストラクチャ ポリシーの名前。
- containerType：コンテナのタイプは APIC であることが必要です。
- applicationProfileName：コンテナの作成に使用するアプリケーション プロファイルの名前。

コード

```
public class APICContainerVirtualInfraPolicyUpdateExample
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
 UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
 APICContainerVirtualInfraStructurePolicy policy = new
APICContainerVirtualInfraStructurePolicy();
 policy.setPolicyName("VIPolicy1");
 policy.setPolicyDescription("VIPolicy1 Description");
 policy.setApplicationProfileName("applicationProfileName");
   /**
   * Update the service Container Virtual Infra Policy
   */
  boolean obj = instance.userAPIUpdateServiceContainerVirtualInfraPolicy(policy);
  System.out.println(obj);
}
}
```

結果

仮想インフラストラクチャ ポリシーが正常に更新された場合、結果は True となります。

実装

userAPIUpdateServiceContainerVirtualInfraPolicy API を呼び出し、更新の必要な詳細内容とともに仮想インフラストラクチャ ポリシー名を渡します。

関連項目

APIC 仮想インフラストラクチャ ポリシーの作成

すべての APIC 仮想インフラストラクチャ ポリシーのリスト化

APIC 仮想インフラストラクチャ ポリシーの取得

APIC 仮想インフラストラクチャ ポリシーの削除

APIC 仮想インフラストラクチャ ポリシーの削除

目標

Cisco UCS Director から APIC 仮想インフラストラクチャ ポリシーを削除します。

コンテキスト

仮想アカウントがマッピングされたグループに属しているユーザは、仮想インフラストラクチャ ポリシーを削除できます。

前提条件

仮想インフラストラクチャ ポリシーが存在している必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=
apic:userAPIDeleteServiceContainerVirtualInfraPolicy&opData={param0:
{"policyName":"apicPolicy”}
```

コンポーネント

- policyName：削除する仮想インフラストラクチャ ポリシーの名前。

コード

```
public class APICContainerVirtualInfraPolicyDeleteExample
{
 public static void main(String[] args) throws Exception {
 CuicServer server =
CuicServer.getAPI("172.22.234.33","EF0FE312B5444E7B8DE4836DDC55F4A0", "https", 443);
 CuicServer server = CuicServer.getAPI("<IP address>", "<REST Key>", "https", 443);
 UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
 APICContainerVirtualInfraStructurePolicy policy = new
APICContainerVirtualInfraStructurePolicy();
 policy.setPolicyName("apicPolicy");

 boolean isDeleted = instance.userAPIDeleteServiceContainerVirtualInfraPolicy(policy);

 System.out.println("is Policy Deleted ?"+isDeleted);
}
}
```

結果

APIC 仮想インフラストラクチャ ポリシーが正常に削除された場合、結果は True となります。

実装

userAPIDeleteServiceContainerVirtualInfraPolicy API を呼び出し、仮想インフラストラクチャ ポリシー名を渡します。

関連項目

APIC 仮想インフラストラクチャ ポリシーの作成

すべての APIC 仮想インフラストラクチャ ポリシーのリスト化

APIC 仮想インフラストラクチャ ポリシーの取得

APIC 仮想インフラストラクチャ ポリシーの変更

サービス コンテナの管理

テンプレートを使用したサービス コンテナの作成

目標

テンプレートを使用してサービス コンテナを作成します。

コンテキスト

VDC または VM をプロビジョニングするテンプレートを使用してコンテナを作成します。

前提条件

- ログイン ユーザにサービス コンテナを作成する権限が必要です。
- コンテナ テンプレートが Cisco UCS Director で使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=fenced:userAPICreateServiceContainerWithoutCatalog&opData=
{param0:"SDKCont",param1:2,param2:"SDKContTemplate"}
```

コンポーネント

コンテナ テンプレートが必要です。

コード

```
public class CreateServiceContainerExample
{
 public static void main(String[] args) throws Exception
 {
 CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
 UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
 int srId = instance.userAPICreateServiceContainerWithoutCatalog("SDKCont", 2,
"SDKContTemplate");
 System.out.println(" Service Container Request id "+srId);
 }
}
```

結果

サービス コンテナを作成するためのサービス リクエスト ID が返されます。

実装

サービス コンテナを作成するには、userAPICreateServiceContainerWithoutCatalog API を呼び出して、コンテナ名、グループ ID およびコンテナ テンプレートを渡します。

関連項目

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

サービス コンテナの取得

目標

Cisco UCS Director からサービス コンテナの詳細を取得します。

コンテキスト

VDC または VM をプロビジョニングするコンテナの可用性を識別します。

前提条件

サービス コンテナの ID が分かっている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=fenced:userAPIGetServiceContainerData&opData={param0:2}
```

コンポーネント

サービス container id：詳細を表示する必要があるサービス コンテナの ID。

コード

```
public class GetServiceContainerDataExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server =
CuicServer.getAPI("192.0.2.207","1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
  ContainerDataObjects containerDataObject = instance.userAPIGetServiceContainerData(2);

  System.out.println("Container Id = "+containerDataObject.getContainerId());
  System.out.println("Gateway  =
"+containerDataObject.getVmRequestBundle().getGateway());
 }
      }
```

結果

特定の ID のサービス コンテナの詳細が返されます。

実装

サービス コンテナの詳細を取得するには、サービス コンテナの ID を渡すことで、userAPIGetServiceContainerData API を呼び出します。

関連項目

テンプレートを使用したサービス コンテナの作成

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

カタログを使用したサービス コンテナの取得

目標

サービス コンテナの作成に使用されるカタログ項目とともにサービス コンテナの詳細を取得します。

コンテキスト

VDC または VM をプロビジョニングするコンテナの可用性を識別します。

前提条件

ログイン ユーザにサービス コンテナの詳細を取得する権限が必要です。

REST の URL

```
/app/api/rest?formatType=json&opName=fenced:userAPIGetServiceContainerDetails&opData={param0:2}
```

コンポーネント

container id：詳細を表示する必要があるサービス コンテナの ID。

コード

```
public class GetServiceContainerDetails
{
 public static void main(String[] args) throws Exception
 {
  CuicAPIClient client = new CuicAPIClient("10.23.210.119", 80,
"38B101A62AF14024964D6A87C23C09DE", "http");
  List listObj = new ArrayList();
  //Paas the container id
  listObj.add("4");
  JsonElement jsonResponse =
client.sendJSONRequest("fenced:userAPIGetServiceContainerDetails", listObj);
  JsonArray array = jsonResponse.getAsJsonObject().get("rows").getAsJsonArray();
  for (int count = 0; count < array.size(); count++)
   {
   JsonElement jsonElement = array.get(count);
   JsonObject jsonObject = jsonElement.getAsJsonObject();
   System.out.println("Overview_Group:  " +
jsonObject.get("Overview_Group").getAsString());
   System.out.println("Overview_ID:  " + jsonObject.get("Overview_ID").getAsString());

   System.out.println("Overview_containerName:  " +
jsonObject.get("Overview_containerName").getAsString());
   System.out.println("Overview_VM_Counts:  " +
jsonObject.get("Overview_VM_Counts").getAsString());
   System.out.println("Overview_ServiceRequestStatus:  "
     + jsonObject.get("Overview_ServiceRequestStatus").getAsString());
   System.out.println("vInfraPolicyInfo_ContainerType:  "
     + jsonObject.get("vInfraPolicyInfo_ContainerType").getAsString());
   System.out.println("Policy_VirtualCompute:  " +
jsonObject.get("Policy_VirtualCompute").getAsString());
   System.out.println("Policy_VirtualNetwork:  " +
jsonObject.get("Policy_VirtualNetwork").getAsString());
   System.out.println("Policy_VirtualSystem:  " +
jsonObject.get("Policy_VirtualSystem").getAsString());
   System.out.println("Policy_VirtualStorage:  " +
jsonObject.get("Policy_VirtualStorage").getAsString());
   }
  }
}
```

結果

次のサービス コンテナの詳細が返されます。

- Overview_Group：Default Group
- Overview_ID：4
- Overview_containerName：cont1
- Overview_VM_Counts：Container is Empty
- Overview_ServiceRequestStatus：Complete
- vInfraPolicyInfo_ContainerType：Fenced Virtual
- Policy_VirtualCompute：VNX_CLOUD69 - Default Computing Policy
- Policy_VirtualNetwork：VNX_CLOUD69 - Default Network Policy
- Policy_VirtualSystem：Default System Policy
- Policy_VirtualStorage：VNX_CLOUD69 - Default Storage Policy

実装

サービス コンテナの詳細を取得するには、コンテナの ID を渡すことで、userAPIGetServiceContainerDetails API を呼び出します。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

Cisco UCS Director でのすべてのサービス コンテナのリスト化

目標

Cisco UCS Director で使用可能なすべてのサービス コンテナを取得して表示します。

コンテキスト

Cisco UCS Director 内のサービス（APIC および Fenced）コンテナのリストを表示します。

前提条件

サービス コンテナが Cisco UCS Director で使用できる必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=apic:userAPIGetAllServiceContainers&opData={}
```

応答


```
{ "serviceResult":[{"id":2,"containerType":"APIC","containerName":"cdevBld",
"containerLabel":"","containerState":2,"tenantIdentity":"VNX_APIC185@cdev23",
"privateNetwork":null,"hostsNumberPerTier":"","groupId":4,"vdcId":2,"provisionedTime":
1464001006693,"termiantionTime":-1,"srId":1107,"enableDR":false,"primaryServiceContainerId":
-1,"serviceContainerRole":"PRIMARY","configureResourceLimit":true,"cpu":1.0,"ncpu":3,
"memory":3.0,"provisionedDiskGBLimit":20.0,"halfWidthPhysicalServerLimit":1,
"fullWidthPhysicalServerLimit":1,"enableNetworkMgmt":true,"networkThrougput":"100M",
"customTierLabels":null,"vmLabels":{"list":[],"moTypeName":"com.cloupia.model.
serviceContainer.VMsLabelCustomizationConfig","validatorName":null},
"customTierProperty":null}], "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPIGetAllServiceContainers" }
```


コンポーネント

なし

コード

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Properties;

import com.cisco.cuic.api.client.APITabularReport;
import com.cisco.cuic.api.client.ContainerVirtualMachine;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.client.JSON;
import com.cisco.cuic.api.models.UserAPIFencedContainer;
import com.cisco.cuic.api.models.UserAPIGlobal;
import
com.cisco.cuic.api.models.servicecontainer.APIFencedVirtualContainerTemplateConfig;
import com.cisco.cuic.api.models.servicecontainer.APIFencedVirtualMachineConfig;
import com.cisco.cuic.api.models.servicecontainer.APIServiceContainerTemplate;
import com.cisco.cuic.api.models.servicecontainer.ContainerDataObjects;
import com.cisco.cuic.api.models.servicecontainer.DFAConfig;
import com.cisco.cuic.api.models.servicecontainer.FencedContainerOptions;
import com.cisco.cuic.api.models.servicecontainer.FencedContainerPolicies;
import com.cisco.cuic.api.models.servicecontainer.FencedContainerWorkflowConfig;
import com.cisco.cuic.api.models.servicecontainer.FencedNetwork;
import com.cisco.cuic.api.models.servicecontainer.FencedVirtualMachineNetworkConfig;
import com.cisco.cuic.api.models.servicecontainer.NetworkConfigProvisionedPortGroup;
import com.cisco.cuic.api.models.servicecontainer.NetworkConfigRequest;
import com.cisco.cuic.api.models.servicecontainer.NetworkConfigResult;
import com.cisco.cuic.api.models.servicecontainer.NetworkTopology;
import com.cisco.cuic.api.models.servicecontainer.OutboundACL;
import com.cisco.cuic.api.models.servicecontainer.OwnerInfo;
import com.cisco.cuic.api.models.servicecontainer.PortMapping;
import com.cisco.cuic.api.models.servicecontainer.ServiceContainer;
import com.cisco.cuic.api.models.servicecontainer.ServiceContainervInfraPolciyDef;
import com.cisco.cuic.api.models.servicecontainer.ServiceContainervInfraPolicy;
import com.cisco.cuic.api.models.servicecontainer.VirtualSwitch;
import com.cloupia.sdk.api.CloupiaClient;
import com.google.gson.JsonObject;
import com.google.gson.JsonParser;

public class TestApplicationContainer
{
 private static final String REST_SERVER_PROPERTIES = "rest-server.properties";
 private static final String UCSM_CONTEXT_NAME = "ucsm";
 private Properties props;
 private CloupiaClient client;
 private Map<String, String> reportLabelIdMap;
 private JsonObject obj;
 private JsonParser parser;

 public TestApplicationContainer()
{
   }

 public static void main(String[] args) throws Exception {
  TestApplicationContainer obj = new TestApplicationContainer();
  CuicServer server = CuicServer.getAPI("172.22.234.243", "
CF87FA987C8F4BBF814F2BB68CA6A823", "https", 443);
  UserAPIFencedContainer fencedC = new UserAPIFencedContainer(server);
  executeGetServiceContainerDetails(fencedC);
 }
 public static void executeGetServiceContainerDetails(UserAPIFencedContainer fencedC)
 throws Exception
{
  APITabularReport atr = new APITabularReport();
  atr = fencedC.userAPIGetServiceContainerDetails(1);

  List<Map<String, Object>> listofMaps = atr.getRows();
  // System.out.println("rows " + atr.getRowCount());
  for (Map<String, Object> map : listofMaps) {
   for (Entry<String, Object> entry : map.entrySet()) {
    // System.out.println(entry.getKey() + ":");
    String o = entry.toString();
```

```
        System.out.println(o);
      }
    }
  }
```

結果

コードによって Cisco UCS Director 内のサービス コンテナのリストが返されます。

実装

実装は必要ありません。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

コンテナ VM への階層の追加

目標

APIC コンテナ内の VM に階層を追加します。

（注） Cisco UCS Director リリース 6.x 以降のリリースでは、userAPIAddTierToContainerVM API は廃止されています。コンテナに階層を追加するには、userAPIAddTierToContainer API を使用できます。詳細については、APIC コンテナへの階層の追加，（71 ページ）を参照してください。

コンテキスト

APIC コンテナ VM に階層を追加します。

前提条件

- APIC コンテナが Cisco UCS Director で使用できる必要があります。
- APIC コンテナで VM が使用可能である必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
apic:userAPIAddTierToContainerVM&opData={param0:{"containerId":2,"tierName":"app",
"tierLabel":"app","isIsolated":false,"parentTierName":""}}
```

応答

```
{ "serviceResult":2197, "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPIAddTierToContainerVM" }
```

コンポーネント

userAPIAddTierToContainerVM API のパラメータは次のとおりです。

- int containerId：APIC コンテナの ID。
- String tierName：APIC コンテナに追加する階層の名前。
- String tierLabel：階層のラベル。
- boolean isIsolated：この階層を APIC コンテナ内の親階層に関連付ける場合、True に設定します。
- String parentTierName：この階層を関連付ける親階層の名前。isIsolated パラメータが True に設定されているとき、親階層の名前を入力します。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.apic.APIAPICAddTierToContainerParams;
import com.cisco.cuic.api.models.apic.APIAPICAddvNICToContainerVMParams;
import com.cisco.cuic.api.models.apic.userAPIContainerandContract;

public class userAPIAddTierToContainerVMTest
{
  public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);
  userAPIContainerandContract instance = new userAPIContainerandContract(api);
  APIAPICAddTierToContainerParams param=new APIAPICAddTierToContainerParams();
  param.setContainerId(2);
  param.setIsolated(false);
  param.setParentTierName("");
  param.setTierLabel("apps");
  param.setTierName("apps");
  int requestId=instance.userAPIAddTierToContainerVM(param);
  System.out.println(requestId);
 }
}
```

結果

APIC コンテナ VM に階層を追加できたときに、サービス リクエスト ID を返します。

実装

userAPIAddTierToContainerVM API を使用して、APIC コンテナ VM に階層を追加します。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

APIC コンテナへの階層の追加

目標

APIC コンテナに階層を追加します。

コンテキスト

APIC コンテナに階層を追加します。userAPIAddTierToContainerVM API は廃止されているため、userAPIAddTierToContainer API を使用することを推奨します。

前提条件

APIC コンテナが Cisco UCS Director で使用可能である必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=apic:userAPIAddTierToContainer&opData=
{param0:{"containerId":1,"tierName":"app","tierLabel":"app","isIsolated":false,
"parentTierName":""}}
```

応答

```
{ "serviceResult":17, "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPIAddTierToContainer" }
```

コンポーネント

- Int containerId：APIC コンテナの ID。
- String tierName：APIC コンテナに追加する階層の名前。
- String tierLabel：階層のラベル。
- boolean isIsolated：この階層を APIC コンテナ内の親階層に関連付ける場合、True に設定します。
- String parentTierName：この階層を関連付ける親階層の名前。isIsolated パラメータが True に設定されているとき、親階層の名前を入力します。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIAPICContainer;
import com.cisco.cuic.api.models.apic.APIAPICAddTierToContainerParams;

public class TestUserAPIAddTierToContainer {

 public static void main(String[] args) {
  CuicServer server = CuicServer.getAPI("172.22.234.172",
"91E7A134048F45FD88A29FCBBFD2C233", "https", 443, 12000, "none", "");
   UserAPIAPICContainer instance = new UserAPIAPICContainer(server);
   APIAPICAddTierToContainerParams apicAddTierToContainerParams = new
APIAPICAddTierToContainerParams();
   apicAddTierToContainerParams.setContainerId(1);
   apicAddTierToContainerParams.setTierName("app");
   apicAddTierToContainerParams.setTierLabel("app");
   apicAddTierToContainerParams.setIsolated(false);
   apicAddTierToContainerParams.setParentTierName("");
   try {
   int requestId = instance.userAPIAddTierToContainer(apicAddTierToContainerParams);
   System.out.println("Service Request Id = "+requestId);
  } catch (Exception e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
  }
 }

}
```

結果

APIC コンテナに正常に階層が追加されると、サービス リクエスト ID が返されます。

実装

userAPIAddvNICToContainerVM API を呼び出して、APIC コンテナ VM に vNIC を追加します。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

サービス コンテナの削除

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

目標

APIC コンテナ内の VM に仮想ネットワーク インターフェイス カード（vNIC）を追加します。

コンテキスト

vNIC を APIC コンテナ VM に追加します。

前提条件

- APIC コンテナが Cisco UCS Director で使用できる必要があります。
- APIC コンテナで VM が使用可能である必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
apic:userAPIAddvNICToContainerVM&opData={param0:{"containerId":"2","vmId":"1",
"vmUsername":"root","vmPassword":"cloupia123","networkName":"Con"}}
```

応答

```
{ "serviceResult":2190, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"apic:userAPIAddvNICToContainerVM" }
```

コンポーネント

userAPIAddvNICToContainerVM API のパラメータは次のとおりです。

- int containerId：APIC コンテナの ID。
- int vmId：vNIC の追加が必要な VM の ID。
- String vmUsername：NetScalar デバイスへのアクセスに使用されるユーザ名。
- String vmPassword：NetScalar デバイスへのアクセスに使用されるパスワード。
- String networkName：VM の存在するのと同じアプリケーション コンテナの階層またはネットワーク。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIVMware;
import com.cisco.cuic.api.models.apic.APIAPICAddvNICToContainerVMParams;
import com.cisco.cuic.api.models.apic.userAPIContainerandContract;

public class userAPIAddvNICToContainerVMTest
{
  public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);
  userAPIContainerandContract instance = new userAPIContainerandContract(api);
  APIAPICAddvNICToContainerVMParams param=new APIAPICAddvNICToContainerVMParams();
  param.setContainerId("2");
  param.setNetworkName("Con");
  param.setVmId("1");
  param.setVmUsername("root");
  param.setVmPassword("cloupia123");
  int requestId=instance.userAPIAddvNICToContainerVM(param);
  System.out.println(requestId);
 }
}
```

結果

APIC コンテナ VM に vNIC を追加できたときに、サービス リクエスト ID を返します。

実装

userAPIAddvNICToContainerVM API を使用して、APIC コンテナ VM に □NIC を追加します。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

サービス コンテナの削除

サービス コンテナの削除

目標

Cisco UCS Director からサービス コンテナを削除します。

コンテキスト

コンテナ内のプロビジョニングされた VM によって使用されるリソースを再利用します。VM を使用すると、VM プロビジョニングがロールバックされます。次に、コンテナが削除され、VM に割り当てられているリソースが解放されます。

前提条件

- ログイン ユーザにサービス コンテナを削除する権限が必要です。
- コンテナに VM がないこと。

REST の URL

```
/app/api/rest?formatType=json&opName=fenced:userAPIDeleteServiceContainer&opData={param0:3}
```

コンポーネント

containerId：削除するサービス コンテナの ID。

コード

```
public class DeleteServiceContainerDataExample
{
 public static void main(String[] args) throws Exception
 {
 CuicServer server = CuicServer.getAPI("192.0.2.207","1A8DE698E2BF4C0B989476A369F0FC64",
 "https",443);
 UserAPIFencedContainer instance = new UserAPIFencedContainer(server);
 int srId = instance.userAPIDeleteServiceContainer(2);
 System.out.println("Delete SR id  "+srId);
 }
}
```

結果

サービス コンテナが削除され、サービス リクエスト ID が返されます。

実装

サービス コンテナ ID を渡すことで、サービス コンテナを削除します。

関連項目

テンプレートを使用したサービス コンテナの作成

サービス コンテナの取得

カタログを使用したサービス コンテナの取得

Cisco UCS Director でのすべてのサービス コンテナのリスト化

コンテナ VM への階層の追加

APIC コンテナへの階層の追加

コンテナ VM への仮想ネットワーク インターフェイス カードの追加

コントラクトの管理

コントラクトの作成

目標

APIC コンテナ内に 2 つのネットワーク間のコントラクトを作成します。

コンテキスト

2 つのネットワーク間のコントラクトを作成します。

前提条件

Cisco UCS Director に APIC コンテナが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
apic:userAPICreateContract&opData={param0:{"containerId":2,"ruleName":"rule1",
"ruleDescription":"sample","sourceNetwork":"web","destNetwork":"app",
"createRule":true,"protocol":"TCP","applyBothDirections":true,"sourcePortStart":"10",
"sourcePortend":"20","destinationPortStart":"30","destinationPortEnd":"40",
"fireallAction":1000,"enableStatefull":true}}
```

応答

```
{ "serviceResult":2166, "serviceError":null, "serviceName":"InfraMgr",
"opName":"apic:userAPICreateContract" }
```

コンポーネント

userAPICreateContract API のパラメータは次のとおりです。

- int containerId：コントラクトが作成される APIC コンテナの一意の ID。
- String ruleName：ルールの名前。ルールは、Cisco UCS Director により自分の参照用として内部的に使用されます。
- String ruleDescription：ルールの説明。APIC 上で作成されるルール フィルタ名は、コンテナ名を使用して自動生成されます。
- String sourceNetwork：コントラクト ルールを適用する送信元ネットワークを選択します。
- String destNetwork：コントラクト ルールを適用する宛先ネットワークを選択します。
- boolean createRule：ルールを作成するには True に設定します。
- String protocol：通信プロトコル。
- boolean applyBothDirections：送信元から宛先までのトラフィックに対して同じコントラクトを適用する場合に、True に設定します。
- int sourcePortStart：送信元に指定するポート範囲の開始ポート番号です。
- int sourcePortend：送信元に指定するポート範囲の終了ポート番号です。
- int destinationPortStart：宛先に指定するポート範囲の開始ポート番号です。
- int destinationPortEnd：宛先に指定するポート範囲の終了ポート番号です。
- int fireallAction：このパラメータは空にしておきます。
- boolean enableStatefull：ステートフルを有効化するには True に設定します。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.apic.APIAPICAddTierToContainerParams;
import com.cisco.cuic.api.models.apic.APICCreateContractParams;
import com.cisco.cuic.api.models.apic.userAPIContainerandContract;

public class userAPICreateContractTest
{
 public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);
  userAPIContainerandContract instance = new userAPIContainerandContract(api);
  APICCreateContractParams param=new APICCreateContractParams();
  param.setContainerId(2);
  param.setApplyBothDirections(true);
  param.setCreateRule(true);
  param.setDestinationPortEnd("40");
  param.setDestinationPortStart("30");
  param.setDestNetwork("app");
  param.setSourceNetwork("web");
  param.setEnableStatefull(true);
  param.setFireallAction(100);
  param.setProtocol("TCP");
  param.setRuleDescription("sdk");
  param.setRuleName("Rule123");
  param.setSourcePortend("20");
  param.setSourcePortStart("10");
  int requestId=instance.userAPICreateContract(param);
  System.out.println(requestId);
 }
}
```

結果

Cisco UCS Director サーバでコントラクトが正常に作成された後、一意のコントラクト ID が返されます。

実装

userAPICreateContract API を使用して、コントラクト作成に必要なデータを渡します。

関連項目

コントラクトの削除

コントラクトの削除

目標

APIC コンテナ内のネットワーク間のコントラクトを削除します。

コンテキスト

カタログはシステム管理者が削除できます。

前提条件

Cisco UCS Director に APIC コンテナが存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
apic:userAPIDeleteContract&opData={param0:{"ruleId":"rule1","srcNetwork":"web",
"destNetwork":"app"}}
```

応答

```
{"serviceResult":2176, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"apic:userAPIDeleteContract" }
```

コンポーネント

userAPIDeleteContract API のパラメータは次のとおりです。

- String ruleName：削除するルールの名前。
- String srcNetwork：コントラクト ルールを削除する送信元ネットワーク。
- String destNetwork：コントラクト ルールを削除する宛先ネットワーク。

コード

```
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.apic.APICCreateContractParams;
import com.cisco.cuic.api.models.apic.APICDeleteContractParams;
import com.cisco.cuic.api.models.apic.userAPIContainerandContract;

public class userAPIDeleteContractTest
{
  public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);
  userAPIContainerandContract instance = new userAPIContainerandContract(api);
  APICDeleteContractParams param=new APICDeleteContractParams();
  param.setDestNetwork("app");
  param.setRuleId("Rule123");
  param.setSrcNetwork("web");
  int requestId=instance.userAPIDeleteContract(param);
  System.out.println(requestId);
 }
}
```

結果

送信元と宛先のネットワーク間でコントラクトの削除が成功したサービス リクエスト ID を返します。

実装

userAPIDeleteContract API をコールし、ルール名、送信元ネットワークと宛先ネットワークを渡して、送信元と宛先ネットワーク間のコントラクトを削除します。

関連項目

コントラクトの作成

仮想マシンの管理

VM のプロビジョニング

目標

標準カタログを使用して、VMware 上に VM をプロビジョニングできます。

コンテキスト

VMware 上に VM をプロビジョニングします。

前提条件

次のものが Cisco UCS Director で使用可能であることを確認します。

- VM クラウド
- VMware システム
- コンピューティング、ネットワーク、およびストレージのポリシー
- VDC
- 標準カタログ

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIProvisionRequest&opData=
{param0:{"catalogName":"catalog_1","vdcName":"vdc_1","userID":"admin","selectedDiskDataStore":
"[{'diskName':'Hard disk 1','diskSize':'10','datastore':'QA_DS01','diskType':'0'},
{'diskName':'Hard disk 2','diskSize':'10','datastore':'QA_DS01','diskType':'0'}]"}}
```

応答

```
{ "serviceResult":52, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPIProvisionRequest" }
```

コンポーネント

次のパラメータは、VM のプロビジョニングで必須です。

- String catalogName：VM プロビジョニングの起点となるカタログの名前。
- String vdcName：VDC の名前。
- String userID：ユーザの一意の ID。

（注） catalogName、vdcName、および userId を使用すると、Cisco UCS Director で定義されている各種のポリシー（VMware システム ポリシー、VMware コンピューティング ポリシー、VMware ストレージ ポリシー、および VMware ネットワーク ポリシー）に基づいて VM が作成されます。

次のパラメータは、VM のプロビジョニングでオプションです。

- int durationHours：VM がアクティブな期間。
- nt beginTime：VM のプロビジョニングが開始される時刻。
- int quantity：プロビジョニングされる VM の数。
- int memoryMB：プロビジョニングされる VM のプライマリ メモリ（GB 単位）。
- int diskGB：プロビジョニングされる VM のハード ディスク（GB 単位）。
- int cores：プロビジョニングされる VM の vCPU の数。
- int estimatedCost：サービス リクエストの推定コスト。
- String comments：VM のプロビジョニングに関するコメント。
- String additionalInfo：VM のプロビジョニングに関する追加情報。
- int chargeFrequency：ユーザに請求する頻度。このパラメータは、Hourly または Monthly の値を受け入れます。
- String nicAliasName：ネットワーク インターフェイス カード（NIC）エイリアスの名前。
- String nicPortGroupName：NIC のポート グループ名。

（注） NIC ポート グループ名の形式は、cloudName@SwitchName@portgroup type@Network name です。たとえば、"nicPortGroupName":"vmware_cloud_82@vSwitch0@Virtual Machine Portgroup@VM Network" となります。

- boolean resourceAllocated：VM にリソースを割り当てる場合は、true に設定します。
- String allocatedHost：VM に割り当てるホスト名。
- String allocatedDataStore：VM に割り当てるデータストア。

- String allocatedResourcePool：VM に割り当てるリソース。
- String altAllocatedHost：VM に割り当てる代替ホスト名。
- String altAllocatedDataStore：VM に割り当てる代替データストア。
- String altAllocatedResourcePool：VM に割り当てる代替リソース プール。
- int customStartupMemory：VM に設定するカスタム起動メモリのサイズ。
- int customMaxMemory：VM に割り当てることができるメモリの最大サイズ。
- int customMemoryBuffer：VM のカスタム メモリ バッファ。
- boolean customMemoryConfig：メモリの構成を許可する場合は、true に設定します。
- String customStoragePolicy：VDC に関して VM をプロビジョニングする際のストレージ ポリシー。ストレージ ポリシーが customStoragePolicy パラメータで定義されている場合、UserAPIProvisionRequest API は VM をプロビジョニングする際にストレージ ポリシーを参照します。ストレージ ポリシーが customStoragePolicy パラメータで定義されていない場合、UserAPIProvisionRequest API は VM をプロビジョニングする際に指定された VDC のデフォルト ストレージ ポリシーを参照します。
- String allocatedCluster：VM に割り当てるクラスタ。
- int customCpuSockets：VM に割り当てる CPU ソケットの数。
- int customCpuCoresPerSocket：ソケットごとに割り当てる CPU の数。
- String altAllocatedCluster：VM に割り当てる代替クラスタ。
- String allocatedAddnlDatastores：VM に割り当てる追加データストア。
- String altAllocatedAddnlDatastores：VM に割り当てる代替追加データストア。
- String altAllocatedAddnlVNICs：VM に割り当てる代替追加 vNIC。
- String altAllocatedAddnlVNICsIpv6：VM に割り当てる、IPv6 をサポートする代替追加 vNIC。

- String selectedDiskDataStore：VMware ストレージ ポリシーで定義されたハードディスクに選択するデータストア。ハード ディスクの名前は、`{'diskName':'diskName','diskSize':'diskSizeinGB','datastore':'datastoreName','diskType':'diskType'}` の形式にする必要があります。有効なディスク タイプは次のとおりです。
  - 0：システム
  - 1：データ
  - 2：データベース
  - 3：ログ
  - 4：スワップ

  2 つのハードディスクを定義する例：`"selectedDiskDataStore":"[{'diskName':'Hard disk 1','diskSize':'5','datastore':'esxi02_boot','diskType':'0'},{'diskName':'Hard disk 2','diskSize':'5','datastore':'esxi02_boot','diskType':'0'}]"`

  この例では、すべてのディスクが単一のデータストアでプロビジョニングされます。複数のディスクを異なるデータストアでプロビジョニングする場合は、ポリシー カタログで [すべてのディスクを単一のデータストアでプロビジョニングする（Provision all disks in single datastore）] チェックボックスをオフにします。
- String actionId：VM のプロビジョニングに使用するワークフロー アクション ID。
- String vmName：プロビジョニングする VM の名前。
- String vdcCategory：VDC のカテゴリ。
- String windowsLicensePool：Windows ライセンス プール。
- String templateUserId：テンプレートのユーザ ID。
- String templatePassword：テンプレートのパスワード。
- String credentialOption：VM アクセス クレデンシャル。
- boolean provisionAllDisk：単一のデータストアですべてのディスクをプロビジョニングする場合は、true に設定します。
- boolean enableGuestCustomization：VM でゲストのカスタマイズを有効にする場合は、true に設定します。
- boolean enablePostProvisioningCustomActions：VM のプロビジョニング後に実行されるオーケストレーション ワークフローを有効にする場合は、true に設定します。
- String workflow：VM プロビジョニングのワークフロー。
- Int vmId：VM のアイデンティティ。
- Int vMAppChargeFrequency：VM アプリケーションに請求する頻度。このパラメータは、Hourly または Monthly の値を受け入れます。
- Int activeVMAppCost：テンプレートに含まれているアプリケーションのコスト。

- Int inactiveVMAppCost：このカタログに対する、非アクティブな状態の VM の毎時または毎月の
コスト。
- boolean useLinkedClone：リンク付き複製機能を使用する場合は、true に設定します。
- Int snapshotId：VM のスナップショット ID。
- String snapshotKey：スナップショットのキー。
- String newSnapshotName：VM の新しいスナップショットの名前。
- boolean isHighlyAvailable：VM で高可用サポートが利用可能な場合は、true に設定します。
- List postProvWFUserInputs：VM のプロビジョニング後に実行するワークフローのユーザ入力のリ
ストを設定します。

コード

```
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.client.APIProvisionParams;
public class TestuserAPIProvisionRequest {
public static void main(String[] args) {
  CuicServer server = CuicServer.getAPI("172.29.110.222",
"6F0063A7F7654561A790EACCE1E8626F", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APIProvisionParams params=new APIProvisionParams();
  int srId;
  params.setCatalogName("catalog_1");
  params.setVdcName("vdc_1");
  params.setUserID("admin");
  String selectedDiskDataStore="[{'diskName':'Hard disk
1','diskSize':'5','datastore':'QA_DS01','diskType':'0'},
{'diskName':'Hard disk 2','diskSize':'-1','datastore':'QA_DS01','diskType':'0'}]";
  params.setSelectedDiskDataStore(selectedDiskDataStore); //Optional
   try {
   srId=instance.userAPIProvisionRequest(params);
   System.out.println("Service Request Id for VM provision request = "+srId);
  } catch (Exception e) {

   e.printStackTrace();
  }
 }
}
```

結果

VM をプロビジョニングするためのサービス リクエスト ID が返されます。

実装

userAPIProvisionRequest API を呼び出して、VM をプロビジョニングするために必要なパラメータを渡します。

関連項目

VM の電源オン

VM の再起動

VM への仮想ネットワーク インターフェイス カードの追加

VM の電源オフ

VM の電源オン

目標

VM の電源をオンにします。

コンテキスト

VDC で使用できる VM を管理します。

前提条件

VM がアクセスに使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIExecuteVMAction&opData={param0:5,
param1:"powerOn",param2:"Power On sample test"}
```

コンポーネント

- int vmId：電源をオンにする仮想マシンの ID。
- String actionName：powerOn として値を設定します。
- String comments：オプション。追加情報（ある場合）。

コード

```
public class userAPIExecuteVMActionExample
{
public static void main(String[] args) throws Exception
{
   CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
   UserAPIGlobal instance = new UserAPIGlobal(server);
   String statusMsg = instance.userAPIExecuteVMAction(1,"powerOn","Testing");
   System.out.println(" Response msg is "+statusMsg);
 }
}
```

結果

電源がオンにされた VM のステータスが表示されます。

実装

VM の電源をオンにするには、VM ID を渡して、userAPIExecuteVMAction API でアクション名を電源オンとして設定します。

関連項目

VM のプロビジョニング

VM の再起動

VM への仮想ネットワーク インターフェイス カードの追加

VM の電源オフ

VM の再起動

目標

VM を再起動します。

コンテキスト

VDC で使用できる VM を管理します。

前提条件

VM がアクセスに使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIExecuteVMAction&opData={param0:5,
param1:"reboot",param2:"Reboot VM"}
```

コンポーネント

- int vmId：再起動する仮想マシンの ID。
- String actionName：reboot として値を設定します。
- String comments：オプション。追加情報（ある場合）。

コード

```
public class userAPIExecuteVMActionExample {
public static void main(String[] args) throws Exception {
CuicServer server = CuicServer.getAPI("192.0.2.207", "1A8DE698E2BF4C0B989476A369F0FC64",
 "https", 443);

  UserAPIGlobal instance = new UserAPIGlobal(server);
  String statusMsg = instance.userAPIExecuteVMAction(1, "Reboot", "Testing");
 System.out.println(" Response msg is "+statusMsg);}
}
```

結果

リブートした VM のステータスが表示されます。

実装

VM を再起動するには、VM ID を渡して、userAPIExecuteVMAction API でアクション名を再起動として設定します。

関連項目

VM のプロビジョニング

VM の電源オン

VM への仮想ネットワーク インターフェイス カードの追加

VM の電源オフ

VM への仮想ネットワーク インターフェイス カードの追加

目標

VMに仮想ネットワークインターフェイスカード（vNIC）を追加してネットワークをブリッジします。APIC コンテナと Fenced コンテナの両方、および標準カタログ VM について、使用可能な vNIC が必要です。

コンテキスト

ネットワークの仮想化。

前提条件

portGroupIdentity、スタティック IP プール、有効なコンテナ テンプレート（APIC または Fenced）、および標準カタログ VM が使用可能であることを確認します。

REST の URL

DHCP が True に設定されたリクエスト

```
app/api/rest?formatType=json&opName=genericvm:
userAPIAddVMNICs&opData={param0:{"vmId":531,"vNICConfig":
[{"portGroupIdentity":"vmware117@172.29.110.75@vSwitch0@VM Network@Virtual
 Machine Portgroup","isDHCP":false,"staticIpPool":"10.10.20.8","subnetMask":
"255.255.255.0","gateway":"10.10.20.1"}]}}
```

DHCP が False に設定されたリクエスト

```
/app/api/rest?formatType=json&opName=genericvm:
userAPIAddVMNICs&opData={param0:{"vmId":49,"vNICConfig":
[{"portGroupIdentity":"vmware117@172.29.110.75@vSwitch0@VM Network@Virtual
Machine Portgroup","isDHCP":false,"staticIpPool":"","subnetMask":"","gateway":""}]}}
```

コンポーネント

- vmId：vNIC の追加が必要な VM の ID。
- vNICConfig：VM に追加される vNIC の構成。次のような VM 構成を設定する必要があります。
  - portGroupIdentity：vNIC に指定されるポートグループの ID。ポート グループのアイデンティティとして許可されている形式は、<cloudName>@<hostName>@<switchName>@<portGroupName>@<portGroupType> です。
  - isDHCP：True に設定して DHCP 構成を有効化します。
  - staticIpPool：isDHCP パラメータが True に設定された場合、このフィールドはオプションです。スタティック IP 設定の IP アドレスです。
  - subnetMask：isDHCP パラメータが True に設定された場合、このフィールドはオプションです。スタティック IP 設定のサブネット マスク アドレスです。
  - gateway：isDHCP パラメータが True に設定された場合、このフィールドはオプションです。スタティック IP 設定のゲートウェイ アドレスです。

コード

```
public static void main(String[] args) throws Exception
{
 CuicServer api = CuicServer.getAPI("172.31.234.172",
"E052D5B0D1BD4B3199DEB36620AA0004", "http", 80);
 UserAPIVMware instance = new UserAPIVMware(api);

 List<VMNICsInputConfig> vNICConfig = new ArrayList<VMNICsInputConfig>();
 VMNICsInputConfig nicConfig = new VMNICsInputConfig();

nicConfig.setPortGroupIdentity("vmware169@172.31.232.176@vSwitch0@ctr-sdk-pg-lan0@Virtual
 Machine Portgroup");
 nicConfig.setDHCP(true);
 nicConfig.setStaticIpPool(null);
 nicConfig.setSubnetMask(null);
 nicConfig.setGateway(null);

 vNICConfig.add(nicConfig);

 APIVMNICInputParams param = new APIVMNICInputParams();
 param.setVmId(38);
 param.setvNICConfig(vNICConfig);
 APIVMNICOutputDetails output = instance.userAPIAddVMNICs(param);
 System.out.println("VM Id: "+output.getVmId());
   for(VMNICOutputDetails details : output.getvNicDetails()){
   System.out.println("Adaptor Name: "+details.getAdapterName());
   System.out.println("MAC Address: "+details.getMacAddress());
   System.out.println("Static IP: "+details.getStaticIpPool());
   System.out.println("Subnet Mask: "+details.getSubnetMask());
   System.out.println("Gateway: "+details.getGateway());
  }
}
```

結果

vNIC が VM に追加されます。

結果サンプル

- VM Id: 38
- Adaptor Name: eth1
- MAC Address: 00:50:56:81:76:ba
- Static IP: null
- Subnet Mask: null
- Gateway: null

REST URL の応答

```
{ "serviceResult":{"vmId":49,"vNicDetails":[{"adapterName":"eth1",
"macAddress":"00:50:56:8b:73:8b","staticIpPool":"10.10.10.0",
"subnetMask":"255.255.255.0","gateway":"10.10.10.1"}]}, "serviceError":null,
 "serviceName":"InfraMgr", "opName":"genericvm:userAPIAddVMNICs" }
```

実装

- DHCP 構成の場合、isDHCP を True に設定します。DHCP が設定されると、IP アドレスが動的に指定されます。
- スタティック設定の場合、isDHCP パラメータを False に設定し、staticIPPool、subnetMask、およびゲートウェイ パラメータに値を指定します。
- 選択した VM がコンテナと関連付けられ、isDHCP パラメータが False に設定されていて、かつ staticIPPool、subnetMask、およびゲートウェイ パラメータが指定されていない場合、システムはコンテナ テンプレートのネットワーク設定をチェックしてスタティック IP 設定を取得します。

関連項目

VM のプロビジョニング

VM の電源オン

VM の再起動

VM の電源オフ

VM の電源オフ

目標

VM の電源をオフにします。

コンテキスト

VDC で使用できる VM を管理します。

前提条件

VM がアクセスに使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIExecuteVMAction&opData={param0:5,
param1:"powerOff",param2:"Power off sample test"}
```

コンポーネント

- int vmId：電源をオフにする仮想マシンの ID。
- String actionName：powerOff として値を設定します。
- String comments：オプション。追加情報（ある場合）。

コード

```
public class userAPIExecuteVMActionExample
{
public static void main(String[] args) throws Exception
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  String statusMsg = instance.userAPIExecuteVMAction(1,"powerOff","Testing");
  System.out.println(" Response msg is "+statusMsg);
 }
}
```

結果

電源がオフにされた VM のステータスが表示されます。

実装

VM の電源をオフにするには、VM ID を渡して、userAPIExecuteVMAction API でアクション名を電源オフとして設定します。

関連項目

VM のプロビジョニング

VM の電源オン

VM の再起動

VM への仮想ネットワーク インターフェイス カードの追加

VMware VM ゲストのセットアップと VIX スクリプトの実行

目標

ターゲット VM で特定の操作（VM の電源オン、VM の電源オフ、VM 名の変更など）を実行するための VMware VM ゲストをセットアップします。

コンテキスト

ターゲット VM で VIX スクリプトを実行します。

前提条件

VMware ツールを VM にインストールする必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=genericvm:userAPIExecuteVIXScript&opData={param0:355,
param1:"root",param2:"cloupia123",param3:"/bin/echo \"Hello\";/bin/echo 'NPROXY=\$1'
"}
```

コンポーネント

VM ID、クレデンシャル、および VIX スクリプトが必要です。

コード

```
public class VIXScriptExecuteUsingGuestSetup
{
 public static void main(String[] args)
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  GuestSetup instance = new GuestSetup(server);
  instance.setVmId(120);
  instance.setCredentialsOptions("Do not Share");
  instance.setUserId("admin");
  instance.setPassword("admin");
  GuestSetupResponse obj = instance.execute();

  System.out.println("userid " + instance.getUserId());
  System.out.println("vm id " + instance.getVmId());
  System.out.println("password " + instance.getPassword());

  ExecuteVIXScript instancevix = new ExecuteVIXScript(server);
  instancevix.setAccountName("cloud123");
  instancevix.setVmId(instance.getVmId());
  instancevix.setCredentialType("Login");
  instancevix.setLogin(instance.getUserId());
  instancevix.setPassword(instance.getPassword());
  instancevix.setScript("/bin/date");
  instancevix.setUndoScript("");
  instancevix.setUndoScriptTask(false);
  instancevix.setOutputDisplay(false);
  ExecuteVIXScriptResponse response = instancevix.execute();
  System.out.println("Respose status Code "+response.getResponseErrorCode());
 }
}
```

結果

応答コードが実行された VIX スクリプトに返されます。

実装

VM で VIX スクリプトを実行するには、userAPIExecuteVIXScript API を使用して、VM ID、クレデンシャル、および VIX スクリプトを渡します。

VMware システム ポリシーの管理

VMware システム ポリシーの作成

目標

VMware システム ポリシーを作成し、使用するテンプレート、タイム ゾーン、OS 情報など、仮想マシン（VM）に対してシステム固有の情報を定義します。

コンテキスト

VM のシステム固有の情報を定義します。

前提条件

なし

REST の URL

このセクションには、さまざまなシナリオで VMware システム ポリシーを作成する REST URL が用意されています。

例 1：Linux イメージで VMware システム ポリシーを作成する

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPICreateVMwareSystemPolicy&opData={param0:{"policyName":"test",
"policyDescription":"sample","vmImageType":"Linux Only","vmNameTemplate":"sample",
"vmnamevalidationPolicy":"sample","hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxVM":{"dnsDomain":"sample",
"dnsSuffix":"sample","dnsServer":"sample","linuxTimeZone":"US/Arizona",
"maxBootWaitTime":10},"windowsVM":"productId":"sample","licenseOwnerName":"sample",
"organizationName":"sample","licenseMode":"sample","noOfLicenseUsers":1000,
"primaryWINs":"sample","secondaryWINS":"sample","maxBootTime":1000,
"isSIDUnique":true,"isAutoLogon":true,"autoLogonCount":1000,
"administratorPassword":"sample","windowsTimezone":"sample","joinTypeDomain":"sample",
"workGroupName":"sample","domain":"sample","domainAdmin":"sample",
"domainPassword":"sample","defineAnnotation":true}}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"genericvm:userAPICreateVMwareSystemPolicy" }
```

例 2：Windows および Linux イメージで VMware システム ポリシーを作成する

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPICreateVMwareSystemPolicy&opData={param0:{"policyName":"test1",
"policyDescription":"sample","vmImageType":"Windows and Linux",
"vmNameTemplate":"sample","vmnamevalidationPolicy":"sample",
"hostNameTemplate":"sample","hostNameValidationPolicy":"sample",
"linuxVM":{"dnsDomain":"sample","dnsSuffix":"sample","dnsServer":"sample",
"linuxTimeZone":"US/Arizona","maxBootWaitTime":10},"windowsVM":
{"productId":"sample","licenseOwnerName":"sample","organizationName":"sample",
"licenseMode":"Per-Seat","noOfLicenseUsers":1000,"primaryWINs":"sample",
"secondaryWINS":"sample","maxBootTime":10,"isSIDUnique":true,
"isAutoLogon":true,"autoLogonCount":1000,"administratorPassword":"sample",
"windowsTimezone":"Alaskan Time","joinTypeDomain":"Domain",
"workGroupName":"sample","domain":"sample","domainAdmin":"sample",
"domainPassword":"sample","defineAnnotation":true}}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"genericvm:userAPICreateVMwareSystemPolicy" }
```

コンポーネント

なし

コード

```
public class CreateSystemPolicyTest
 {
 public static void main(String[] args)throws Exception{
 CuicServer api = CuicServer.getAPI("172.29.110.194",
"6BF80FA2C71E4844AFEB3877CFD60621", "http", 80);
 UserAPIVMware instance = new UserAPIVMware(api);
 ServiceDeliveryPolicyRequestParam service = new ServiceDeliveryPolicyRequestParam();

 LinuxVMParams linux=new LinuxVMParams();
 linux.setDnsDomain("testdomain");
 linux.setLinuxTimeZone("US/Pacific");
 linux.setLinuxVMmaxBootWaitTime(2);
 WindowsVMParams window=new WindowsVMParams();
 window.setOrganizationName("testOrganization");
 service.setPolicyName("test2");
 service.setPolicyDescription("test");
 service.setVmImageType("Linux Only");
 service.setHostNameTemplate("sample1");
 service.setLinuxVM(linux);
 service.setWindowsVM(window);
 boolean policy= instance.userAPICreateVMwareSystemPolicy(service);
 System.out.println(policy);
 }
```

結果

VMware システム ポリシーが正常に作成された場合、結果が True になります。

実装

userAPICreateVMwareSystemPolicy API を使用し、名前と値の形式でシステム固有情報を渡して、VMware システム ポリシーを作成します。

関連項目

VMware システム ポリシー詳細の取得

VMware システム ポリシーの変更

VMware システム ポリシーの削除

VMware システム ポリシー詳細の取得

目標

VMware システム ポリシーの詳細情報を取得します。

コンテキスト

VMware システム ポリシーの詳細は、システム管理者またはエンド ユーザによって取得されます。

前提条件

なし

REST の URL

このセクションには、さまざまなシナリオで VMware システム ポリシーを取得する REST URL が用意されています。

例 1：特定の VMware システム ポリシーの詳細情報を取得する

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPIGetVMwareSystemPolicy&opData=param0:{"policyName":"test",
"policyDescription":"sample","vmImageType":"sample","vmNameTemplate":
"sample","vmnamevalidationPolicy":"sample","hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxVM":{"dnsDomain":"sample",
"dnsSuffix":"sample","dnsServer":"sample","linuxTimeZone":"sample",
"maxBootWaitTime":0},"windowsVM":{"productId":"sample","licenseOwnerName":
"sample","organizationName":"sample","licenseMode":"sample",
"noOfLicenseUsers":1000,"primaryWINs":"sample","secondaryWINS":"sample",
"maxBootTime":1000,"isSIDUnique":true,"isAutoLogon":true,"autoLogonCount":1000,
"administratorPassword":"sample","windowsTimezone":"sample","
joinTypeDomain":"sample","workGroupName":"sample","domain":"sample",
"domainAdmin":"sample","domainPassword":"sample","defineAnnotation":true}}}
```

応答

```
{ "serviceResult":{"policyId":8,"policyName":"test",
"policyDescription":"sample","vmNameTemplate":"sample",
"vmNameValidationPolicy":"sample","isAllowEndUserSuffix":false,"powerOn":true,
"hostNameTemplate":"sample","hostNameValidationPolicy":"sample",
"linuxTimezone":"US/Arizona","linuxVMMaxBootTime":4,"dnsDomain":"sample",
"dnsSuffix":null,"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,
"resourcePool":null,"imageType":1,"windowsLabel":null,"productId":null,
"fullName":"CompanyFullName","orgName":"CompanyName","licenseMode":"Per-Seat",
"licenseUsers":5,"winList":null,"secondaryWINS":null,"windowsVMMaxBootTime":10,
"isSIDUnique":false,"isAutoLogon":true,"autoLogonCount":5,"password":null,
"windowsTimezone":4,"joinTypeDomain":true,"workGroupName":null,"domain":null,
"domainAdmin":null,"domainPassword":null,"defineAnnotation":false,"notes":"",
"customAttributes":{"list":[],"moTypeName":
"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false}, "serviceError":null,
 "serviceName":"InfraMgr", "opName":"genericvm:userAPIGetVMwareSystemPolicy" }
```

例 2：すべての VMware システム ポリシーの詳細情報を取得する

要求

```
/app/api/rest?formatType=json&opName=
genericvm:userAPIGetAllVmwareSystemPolicies&opData={}
```

応答


```
{ "serviceResult":[{"policyId":1,"policyName":"Default System Policy",
"policyDescription":"","vmNameTemplate":"vm-${GROUP_NAME}-SR${SR_ID}",
"vmNameValidationPolicy":"","isAllowEndUserSuffix":false,"powerOn":true,
"hostNameTemplate":"${VMNAME}","hostNameValidationPolicy":"",
"linuxTimezone":"US/Pacific","linuxVMMaxBootTime":10,"dnsDomain":"sdk","dnsSuffix":null,
"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,"resourcePool":"",
"imageType":1,"windowsLabel":null,"productId":"","fullName":"CompanyFullName",
"orgName":"CompanyName","licenseMode":"Per-Seat","licenseUsers":5,"winList":"",
"secondaryWINS":"","windowsVMMaxBootTime":10,"isSIDUnique":false,"isAutoLogon":true,
"autoLogonCount":5,"password":"","windowsTimezone":4,"joinTypeDomain":true,
"workGroupName":"","domain":"","domainAdmin":"","domainPassword":"",
"defineAnnotation":false,"notes":null,"customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":2,
"policyName":"sample","policyDescription":"sample","vmNameTemplate":"sample",
"vmNameValidationPolicy":"sample","isAllowEndUserSuffix":false,"powerOn":true,
"hostNameTemplate":"host","hostNameValidationPolicy":"sample","linuxTimezone":"US/Arizona",
"linuxVMMaxBootTime":4,"dnsDomain":"sample","dnsSuffix":null,"dnsSuffixList":null,
"dnsServer":null,"dnsServerList":null,"resourcePool":null,"imageType":1,"windowsLabel":null,
"productId":null,"fullName":"CompanyFullName","orgName":"CompanyName","licenseMode":
"Per-Seat","licenseUsers":5,"winList":null,"secondaryWINS":null,"windowsVMMaxBootTime":10,
"isSIDUnique":false,"isAutoLogon":true,"autoLogonCount":5,"password":null,"windowsTimezone":4,
"joinTypeDomain":true,"workGroupName":null,"domain":null,"domainAdmin":null,"domainPassword":null,
"defineAnnotation":false,"notes":"","customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":3,"policyName":"sample1",
"policyDescription":"sample","vmNameTemplate":"sample","vmNameValidationPolicy":"sample",
"isAllowEndUserSuffix":false,"powerOn":true,"hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxTimezone":"US/Pacific","linuxVMMaxBootTime":10,
"dnsDomain":null,"dnsSuffix":null,"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,
"resourcePool":null,"imageType":0,"windowsLabel":null,"productId":"sample","fullName":"sample",
"orgName":"","licenseMode":"Per-Server","licenseUsers":1000,"winList":"sample",
"secondaryWINS":"sample","windowsVMMaxBootTime":10,"isSIDUnique":true,"isAutoLogon":true,
"autoLogonCount":1000,"password":"c2FtcGxl","windowsTimezone":2,"joinTypeDomain":true,
"workGroupName":null,"domain":"admin","domainAdmin":"admin","domainPassword":"YWRtaW4=",
"defineAnnotation":false,"notes":"","customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":6,"policyName":"sample5",
"policyDescription":"sample","vmNameTemplate":"sample","vmNameValidationPolicy":"sample",
"isAllowEndUserSuffix":false,"powerOn":true,"hostNameTemplate":"sample","hostNameValidationPolicy":
"sample","linuxTimezone":"US/Arizona","linuxVMMaxBootTime":6,"dnsDomain":"sample",
"dnsSuffix":null,"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,"resourcePool":null,
"imageType":1,"windowsLabel":null,"productId":null,"fullName":"CompanyFullName",
"orgName":"CompanyName","licenseMode":"Per-Seat","licenseUsers":5,"winList":null,
"secondaryWINS":null,"windowsVMMaxBootTime":10,"isSIDUnique":false,"isAutoLogon":true,
"autoLogonCount":5,"password":null,"windowsTimezone":4,"joinTypeDomain":true,"workGroupName":null,
"domain":null,"domainAdmin":null,"domainPassword":null,"defineAnnotation":false,"notes":"",
"customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":7,"policyName":"sample6",
"policyDescription":"sample","vmNameTemplate":"sample","vmNameValidationPolicy":"sample",
"isAllowEndUserSuffix":false,"powerOn":true,"hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxTimezone":"US/Pacific","linuxVMMaxBootTime":10,
"dnsDomain":null,"dnsSuffix":null,"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,
"resourcePool":null,"imageType":0,"windowsLabel":null,"productId":"sample","fullName":"sample",
"orgName":"sample","licenseMode":"Per-Seat","licenseUsers":1000,"winList":"sample",
"secondaryWINS":"sample","windowsVMMaxBootTime":10,"isSIDUnique":true,"isAutoLogon":true,
"autoLogonCount":1000,"password":"c2FtcGxl","windowsTimezone":1,"joinTypeDomain":true,
"workGroupName":null,"domain":"sam","domainAdmin":"sample","domainPassword":"c2FtcGxl",
"defineAnnotation":false,"notes":"","customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":8,
"policyName":"test","policyDescription":"sample","vmNameTemplate":"sample",
"vmNameValidationPolicy":"sample","isAllowEndUserSuffix":false,"powerOn":true,
"hostNameTemplate":"sample","hostNameValidationPolicy":"sample",
"linuxTimezone":"US/Arizona","linuxVMMaxBootTime":4,"dnsDomain":"sample","dnsSuffix":null,
```

```
"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,"resourcePool":null,"imageType":1,
"windowsLabel":null,"productId":null,"fullName":"CompanyFullName","orgName":"CompanyName",
"licenseMode":"Per-Seat","licenseUsers":5,"winList":null,"secondaryWINS":null,
"windowsVMMaxBootTime":10,"isSIDUnique":false,"isAutoLogon":true,"autoLogonCount":5,
"password":null,"windowsTimezone":4,"joinTypeDomain":true,"workGroupName":null,"domain":null,
"domainAdmin":null,"domainPassword":null,"defineAnnotation":false,"notes":"",
"customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false},{"policyId":9,"policyName":"test1"
"policyDescription":"sample","vmNameTemplate":"sample","vmNameValidationPolicy":"sample",
"isAllowEndUserSuffix":false,"powerOn":true,"hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxTimezone":"US/Pacific","linuxVMMaxBootTime":10,
"dnsDomain":null,"dnsSuffix":null,"dnsSuffixList":null,"dnsServer":null,"dnsServerList":null,
"resourcePool":null,"imageType":0,"windowsLabel":null,"productId":"sample","fullName":"sample"
"orgName":"sample","licenseMode":"Per-Seat","licenseUsers":1000,"winList":"sample",
"secondaryWINS":"sample","windowsVMMaxBootTime":10,"isSIDUnique":true,"isAutoLogon":true,
"autoLogonCount":1000,"password":"c2FtcGxl","windowsTimezone":3,"joinTypeDomain":true,
"workGroupName":null,"domain":"sample","domainAdmin":"sample","domainPassword":"c2FtcGxl",
"defineAnnotation":false,"notes":"","customAttributes":{"list":[],
"moTypeName":"com.cloupia.service.cIM.inframgr.profiles.PrivateCloudSystemPolicyAnnotation",
"validatorName":null},"customAttributesList":[],"vmAnnotationsClob":null,
"isUseLicenseFromTemplate":false,"skipCustomization":false}], "serviceError":null,
 "serviceName":"InfraMgr", "opName":"genericvm:userAPIGetAllVmwareSystemPolicies" }
```


コンポーネント

なし

コード

```
public class GetSystemPolicyTest
{
 public static void main(String[] args)throws Exception {
 CuicServer api = CuicServer.getAPI("172.29.110.194",
"6BF80FA2C71E4844AFEB3877CFD60621", "http", 80);
 UserAPIVMware instance = new UserAPIVMware(api);
 ServiceDeliveryPolicyRequestParam service = new ServiceDeliveryPolicyRequestParam();

 service.setPolicyName("test2");
 PrivateCloudSystemProfile policy= instance.userAPIGetVMwareSystemPolicy(service);
 System.out.println("policyName:" + policy.getPolicyName());
 System.out.println("policyDescription:" +policy.getPolicyDescription());
 System.out.println("ImageType:" + policy.getImageType());
 System.out.println("maxBootTime :" + policy.getLinuxVMMaxBootTime());
}
```

結果

有効なシステム ポリシー名を指定すると、VMware システム ポリシーの詳細が正常に取得されます。

userAPIGetVMwareSystemPolicy を呼び出したときに、次の VMware システム ポリシーの詳細が取得されます。

- policyName:test
- policyDescription: sample
- imageType:Linux Only
- maxBootTime: 10

実装

VMware システム ポリシー名を渡すことで userAPIGetVMwareSystemPolicy API を呼び出し、VMware システム ポリシーの詳細を取得します。userAPIGetAllVmwareSystemPolicies API を呼び出して、すべての VMware システム ポリシーの詳細を取得します。

関連項目

VMware システム ポリシーの作成

VMware システム ポリシーの変更

VMware システム ポリシーの削除

VMware システム ポリシーの変更

目標

VMware システム ポリシーを更新します。

コンテキスト

VM のシステム固有の情報を更新します。

前提条件

なし

REST の URL

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPIUpdateVMwareSystemPolicy&opData={param0:{"policyName":"test",
"policyDescription":"sample","vmImageType":"Linux Only",
"vmNameTemplate":"sample","vmnamevalidationPolicy":"sample",
"hostNameTemplate":"sample","hostNameValidationPolicy":"sample","linuxVM":
{"dnsDomain":"sample","dnsSuffix":"sample","dnsServer":"sample",
"linuxTimeZone":"US/Arizona","maxBootWaitTime":4},"windowsVM":
{"productId":"sample","licenseOwnerName":"sample","organizationName":
"sample","licenseMode":"sample","noOfLicenseUsers":1000,"primaryWINs":
"sample","secondaryWINS":"sample","maxBootTime":1000,"isSIDUnique":true,
"isAutoLogon":true,"autoLogonCount":1000,"administratorPassword":"sample",
"windowsTimezone":"sample","joinTypeDomain":"sample","workGroupName":"sample",
"domain":"sample","domainAdmin":"sample","domainPassword":"sample",
"defineAnnotation":true}}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"genericvm:userAPIUpdateVMwareSystemPolicy" }
```

コンポーネント

なし

コード

```
public static void main(String[] args)throws Exception
{
  CuicServer api = CuicServer.getAPI("172.29.110.194",
"6BF80FA2C71E4844AFEB3877CFD60621", "http", 80);
  UserAPIVMware instance = new UserAPIVMware(api);
  ServiceDeliveryPolicyRequestParam service = new ServiceDeliveryPolicyRequestParam();

  service.setPolicyName("test2");
  service.setVmImageType("Linux Only");
  service.setHostNameTemplate("sample2");
  LinuxVMParams linux=new LinuxVMParams();
  linux.setDnsDomain("testdomain");
  linux.setLinuxTimeZone("US/Pacific");
  linux.setLinuxVMmaxBootWaitTime(2);
  WindowsVMParams window=new WindowsVMParams();
  window.setOrganizationName("testOrganization");
  service.setLinuxVM(linux);
  service.setWindowsVM(window);
  boolean policy= instance.userAPIUpdateVMwareSystemPolicy(service);
  System.out.println(policy);
  }
```

結果

VMware システム ポリシーが正常に更新された場合、結果が True になります。

実装

userAPIUpdateVMwareSystemPolicy API を使用し、名前と値の形式で更新の必要なシステム固有情報を渡して、VMware システム ポリシーを更新します。

関連項目

VMware システム ポリシーの作成

VMware システム ポリシー詳細の取得

VMware システム ポリシーの削除

VMware システム ポリシーの削除

目標

ポリシー名を渡すことで、VMware システム ポリシーを削除します。

コンテキスト

VMware システム ポリシーはシステム管理者が削除できます。

前提条件

削除する VMware システム ポリシーが存在する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPIDeleteVMwareSystemPolicy&opData={param0:{"policyName":"test",
"policyDescription":"sample","vmImageType":"sample","vmNameTemplate":
"sample","vmnamevalidationPolicy":"sample","hostNameTemplate":"sample",
"hostNameValidationPolicy":"sample","linuxVM":{"dnsDomain":"sample",
"dnsSuffix":"sample","dnsServer":"sample","linuxTimeZone":"sample",
"maxBootWaitTime":0},"windowsVM":{"productId":"sample","licenseOwnerName":
"sample","organizationName":"sample","licenseMode":"sample",
"noOfLicenseUsers":1000,"primaryWINs":"sample","secondaryWINS":"sample",
"maxBootTime":1000,"isSIDUnique":true,"isAutoLogon":true,"autoLogonCount":1000,
"administratorPassword":"sample","windowsTimezone":"sample","joinTypeDomain":
"sample","workGroupName":"sample","domain":"sample","domainAdmin":"sample",
"domainPassword":"sample","defineAnnotation":true}}}
```

応答

```
{ "serviceResult":true, "serviceError":null, "serviceName":"InfraMgr",
 "opName":"genericvm:userAPIDeleteVMwareSystemPolicy" }
```

コンポーネント

なし

コード

```
public class DeleteSystemPolicyTest
{
 public static void main(String[] args) throws Exception]
 {
 CuicServer api = CuicServer.getAPI("172.29.110.194",
"6BF80FA2C71E4844AFEB3877CFD60621", "http", 80);
 UserAPIVMware instance = new UserAPIVMware(api);
 ServiceDeliveryPolicyRequestParam service = new ServiceDeliveryPolicyRequestParam();

 service.setPolicyName("test23423");
 boolean policy= instance.userAPIDeleteVMwareSystemPolicy(service);
 system.out.println(policy);
}
```

結果

VMware システム ポリシーが正常に削除された場合、結果が True になります。

実装

削除するポリシー名を渡すことで、userAPIDeleteVMwareSystemPolicy API を呼び出し、既存の VMware システム ポリシーを削除します。

関連項目

VMware システム ポリシーの作成

VMware システム ポリシー詳細の取得

VMware システム ポリシーの変更

VMware スナップショットの削除

目標

VMware スナップショットを削除します。

コンテキスト

新しいスナップショット用により多くのディスク領域を提供します。VMwareスナップショットは、管理者のみが削除できます。

前提条件

VMware スナップショットが使用可能である必要があります。

REST の URL

N/A

コンポーネント

スナップショット名は必須です。

コード

```
public class DeleteVMSnapshotExample
{
 public static void main(String[] args)
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  DeleteVMSnapshot instance = new DeleteVMSnapshot(server);
  instance.setVmId(168);
  instance.setSnapshotName("snapshot-2015-01-10");
  instance.setDeleteChild(false);
  DeleteVMSnapshotResponse response = instance.execute();
  System.out.println(" Deleted response "+response.getStatus());
  }
}
```

結果

VM スナップショットが正常に削除された場合、結果は true となります。

実装

実装は必要ありません。

ワークフロー オーケストレーションの管理

Service Request の利用

目標

VM をプロビジョニングする VDC を選択するなど、リソースで一連の操作を実行するためのワークフローを実行するためのサービス要求を送信します。

コンテキスト

一連のタスクを実行するためのワークフローを実行します。

前提条件

ワークフローが Cisco UCS Director で使用できる必要があります。ユーザは、ワークフローを実行する権限を持っている必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPISubmitServiceRequest&opData=
{param0:"testCatalog",param1:"vdc1",param2:1,param3:-1,param4:1,param5:"provisioning
vm"}
```

コンポーネント

ワークフロー名は必須です。

コード

```
public class UserAPISubmitServiceRequestExample
{
 public static void main(String[] args) throws Exception{
 CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
 UserAPIGlobal instance = new UserAPIGlobal(server);
 int srId = instance.userAPISubmitServiceRequest("testCatalog", "vdc", 1, -1, 1,
"UCSD-5.4.0.0");
 System.out.println("srId "+srId);
     }
}
```

結果

サービス リクエスト ID が返されます。

実装

ワークフローを実行するには、userAPISubmitServiceRequest API を呼び出して、ワークフロー名を渡します。

関連項目

VApp リクエストの送信

サービス リクエストの出力の取得

ワークフローのロールバック

VApp リクエストの送信

目標

詳細カタログ タイプおよび引数とともにサービス要求を送信します。

コンテキスト

詳細カタログ タイプ用のワークフローを実行します。

前提条件

詳細カタログが Cisco UCS Director で使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPISubmitVAppServiceRequest&opData={param0:
"PjaCat",param1:{"list":[{"name":"Tenant Name","value":"Pja_27"},{"name":"Tenant
Description",
"value":"none"},{"name":"MSP Admin","value":"asa"},{"name":"MSP Admin
Password","value":"asa"},
{"name":"MSP Admin Email","value":"asa"},{"name":"Datastore Size(GB)","value":"10"},
{"name":"Memory Reservation(MB)","value":"100"},{"name":"No of
CPU","value":"5"},{"name":
"No of VDCs","value":"5"},{"name":"L2 Or L3 External Network
Configuration","value":"None"},
{"name":"L2 VLAN ID","value":""},{"name":"L2 IP Subnet (x.x.x.x/n)","value":""},{"name":
"Tenant IP Subnet (x.x.x.x/n)","value":"10.12.18.0/16"},{"name":"Replication
Required","value":
"No"}]}}
```

コンポーネント

詳細カタログ名

コード

```
public class UserAPISubmitVAppServiceRequestExample
{
 public static void main(String[] args) throws Exception
{
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  APINameValueList list = new APINameValueList();
  APINameValue nv = new APINameValue();
  nv.setName("Tenant Name");
  nv.setValue("Tenant1");
  nv.setName("Tenant Description");
  nv.setValue("");
  nv.setName("Tenant Group");
  nv.setValue("MSPGroup");
  list.addNameValue(nv);
  int srId = instance.userAPISubmitVAppServiceRequest("ExecuteAdvanceCat", list);
  System.out.println("srId "+srId);
  }
}
```

結果

サービス リクエスト ID が返されます。

実装

詳細カタログ タイプ向けのワークフローを実行するには、userAPISubmitVAppServiceRequest API を呼び出して、詳細カタログ名を渡します。

関連項目

Service Request の利用

サービス リクエストの出力の取得

ワークフローのロールバック

サービス リクエストの出力の取得

目標

有効なサービス リクエスト ID を渡すことで、サービス リクエストの出力の詳細を取得します。

コンテキスト

サービス リクエストの出力を確認します。

前提条件

ワークフローの正常に実行されたサービス リクエスト ID が Cisco UCS Director で使用できる必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=servicerequest:
userAPIGetServiceRequestOutputDetails&opData={param0:1}
```

応答

```
{ "serviceResult":{"workflowOutputDetails":[]}, "serviceError":null,
"serviceName":"InfraMgr",
 "opName":"servicerequest:userAPIGetServiceRequestOutputDetails" }
```

コンポーネント

param0：出力を表示させるサービス リクエストの ID。

コード

```
import java.util.List;

import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIServiceRequest;
import com.cisco.cuic.api.models.UserAPIVMware;
import com.cisco.cuic.api.models.servicerequest.APIWorkflowOutputDetails;
import com.cisco.cuic.api.models.servicerequest.APIWorkflowOutputFieldDetails;
import com.cisco.cuic.api.models.vmware.ServiceDeliveryPolicyRequestParam;

public class ServiceRequestOutputDetailsTest {

 public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);

  UserAPIServiceRequest instance = new UserAPIServiceRequest(api);
  APIWorkflowOutputDetails outputDetails =
instance.userAPIGetServiceRequestOutputDetails(1);
  List<APIWorkflowOutputFieldDetails> outputList
=outputDetails.getWorkflowOutputDetails();

  String outputFieldName=outputList.get(0).getOutputFieldName();
  String outputFieldType=outputList.get(0).getOutputFieldType();
  String ouputFieldDescription=outputList.get(0).getOutputFieldDescription();
  String ouputFieldValue=outputList.get(0).getOutputFieldValue();

  System.out.println(outputFieldName);
  System.out.println(outputFieldType);
  System.out.println(outputFieldType);
  System.out.println(ouputFieldValue);
 }
 }
```

結果

サービス リクエストの出力の詳細が表示されます。

実装

userAPIGetServiceRequestOutputDetails API を呼び出し、サービス リクエスト ID を渡して、そのサービス リクエストの出力を表示します。

関連項目

Service Request の利用

VApp リクエストの送信

ワークフローのロールバック

ワークフローのロールバック

目標

特定の操作を元に戻すためにワークフローをロールバックします。ワークフローのロールバックに成功すると、サービス リクエスト ID が生成されます。システム管理者またはエンド ユーザが、ワークフローをロールバックできます。エンド ユーザは、ユーザ ロールに Write - Group Service Request ユーザ権限がある場合にのみサービス リクエストをロールバックできます。

あるユーザが別のユーザによって開始されたサービス リクエストをロールバックする場合、サービス リクエストの開始ユーザから承認を得るためのロールバック ワークフロー承認が起動されます。ロールバック ワークフローは、サービス リクエストの開始ユーザから承認が得られた後に実行されます。

（注） 1 つ以上の複合ワークフロー タスクを持つワークフローがロールバックされる場合でも、ロールバック サービス リクエスト ID は 1 つのみ生成されます。複合ワークフロー タスクに対する子サービス リクエストは生成されません。

コンテキスト

ワークフローの操作を元に戻します。

前提条件

ワークフローの正常に実行されたサービス リクエスト ID が Cisco UCS Director で使用できる必要があります。

REST の URL

```
/app/api/rest?formatType=json&opName=userAPIRollbackWorkflow&opData={param0:40}
```

コンポーネント

int srId：ロールバックするワークフローのサービス リクエスト ID。

コード

```
public class UserAPIRollbackWorkflowExample
{
 public static void main(String[] args) throws Exception
 {
  CuicServer server = CuicServer.getAPI("192.0.2.207",
"1A8DE698E2BF4C0B989476A369F0FC64", "https", 443);
  UserAPIGlobal instance = new UserAPIGlobal(server);
  int srId = instance.userAPIRollbackWorkflow(123);
  System.out.println("srId " + srId);
  }
}
```

結果

サービス リクエスト ID が返されます。

実装

サービス要求をロールバックするには、userAPIRollbackWorkflow API を呼び出して、サービス リクエスト ID を渡します。

関連項目

Service Request の利用

VApp リクエストの送信

サービス リクエストの出力の取得

ワークフローのフィールドの取得

カタログに関連付けられたワークフローの入力フィールドの取得

目標

詳細カタログに関連付けられたワークフローの入力フィールドを取得します。ワークフローの入力ラベル、名前、説明、入力タイプ、フィールド タイプ、およびカタログ タイプなどの入力フィールドを表示できます。

コンテキスト

詳細カタログに関連付けられたワークフローの入力フィールドを確認します。

前提条件

カタログがワークフローに関連付けられている必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=catalog:
userAPIGetCatalogInputDefinition&opData={param0:{"catalogName":"AdvCatalog"}}
```

応答

```
{ "serviceResult":{"details":[{"name":"input_0_input1989","label":"input1",
"description":"","type":"gen_text_input","catalogType":null,"isOptional":false,
"inputFieldType":"text","isMultiSelect":false,"inputFieldValidator":null}]},
 "serviceError":null, "serviceName":"InfraMgr",
"opName":"catalog:userAPIGetCatalogInputDefinition" }
```

コンポーネント

catalogName：詳細カタログの名前。

コード

```
public class GetCatalogInputDef
{
  public static void main(String[] args) throws Exception {
  CuicServer api = CuicServer.getAPI("10.23.210.42", "3E17CFFBA7A64C71B8958F40DE2EC9B3",
 "http", 80);
  UserAPICatalog catalog = new UserAPICatalog(api);
  APICatalogParams params = new APICatalogParams();
  params.setCatalogName("AdvCatalog");
  APIWorkflowInputDetails details = catalog.userAPIGetCatalogInputDefinition(params);

  List<APIWorkflowInputDetail> list = null;
  if(details != null){
   list = details.getDetails();
  }else{
   throw new Exception("No input defination found!");
  }
  if(list != null && list.size() > 0){
  for(int i=0;i<list.size();i++){
   System.out.println("Field Name: "+list.get(i).getName());
   System.out.println("Field Label: "+list.get(i).getLabel());
   System.out.println("Field Description: "+list.get(i).getDescription());
   System.out.println("Input Type: "+list.get(i).getType());
   System.out.println("Field Type: "+list.get(i).getInputFieldType());
   System.out.println("Catalog Type: "+list.get(i).getCatalogType());
  }
  }else{
   System.out.println("No catalog input defination found!");
  }
 }
}
```

結果

カタログに関連付けられたワークフローの入力フィールドがリストされます。

結果サンプル

- Field Name: input_0_input1989
- Field Label: input1
- Field Description:
- Input Type: gen_text_input
- Field Type: text
- Catalog Type: null

実装

userAPIGetCatalogInputDefinition API を使用し、ワークフローに関連付けられた詳細カタログの名前を渡して、ワークフローの入力フィールドを表示します。

関連項目

- カタログに関連付けられたワークフローの出力フィールドの取得, （111 ページ）
- ワークフローの入力フィールドの取得, （113 ページ）
- ワークフローの出力フィールドの取得, （116 ページ）

カタログに関連付けられたワークフローの出力フィールドの取得

目標

詳細カタログに関連付けられたワークフローの出力フィールドを取得します。ワークフロー出力ラベル、名前、説明、およびワークフローのタイプなどの出力フィールドを表示できます。

コンテキスト

詳細カタログに関連付けられたワークフローの出力フィールドを確認します。

前提条件

カタログがワークフローに関連付けられている必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=catalog:userAPIGetCatalogOutputDefinition&opData=
{param0:{"catalogName":"AdvCatalog"}}
```

応答

```
{
"serviceResult":{"workflowOutputFieldList":[{"outputFieldLabel":”name”,"outputFieldName":
"output_0_output1534","outputFieldType":"gen_text_input","outputFieldDescription":""}]},
"serviceError":null, "serviceName":"InfraMgr",
"opName":"catalog:userAPIGetCatalogOutputDefinition" }
```

コンポーネント

catalogName：詳細カタログの名前。

コード

```
public class GetCatalogOutput
{
  public static void main(String[] args) throws Exception
{
  CuicServer api = CuicServer.getAPI("172.29.110.241",
"3E17CFFBA7A64C71B8958F40DE2EC9B3", "http", 80);
  UserAPICatalog catalog = new UserAPICatalog(api);
  APICatalogParams params = new APICatalogParams();
  params.setCatalogName("AdvCatalog");
  APIWorkflowOutputFieldDefinitionList def =
catalog.userAPIGetCatalogOutputDefinition(params);
  List<APIWorkflowOutputFieldDefinition> list;
  if(def != null){
   list = def.getWorkflowOutputFieldList();
  }else{
   throw new Exception("No output defination found!");
  }
    if(list != null && list.size() > 0){
      for(int i=0;i< list.size();i++){
       System.out.println("Field label: "+list.get(i).getOutputFieldLabel());
       System.out.println("Field name: "+list.get(i).getOutputFieldName());
       System.out.println("Field description:
"+list.get(i).getOutputFieldDescription());
       System.out.println("Field type: "+list.get(i).getOutputFieldType());
      }
  }else{
   throw new Exception("No workflow output field found!");
  }
  //System.out.println("lisst = " + list.getWorkflowOutputFieldList().get(0));
  }
}
```

結果

カタログに関連付けられたワークフローの出力フィールドがリストされます。

結果サンプル：

- Field label: output1
- Field name: output_0_output1534
- Field description:
- Field type: gen_text_input

実装

userAPIGetCatalogOutputDefinition API を使用し、ワークフローに関連付けられた詳細カタログの名前を渡して、ワークフローの出力フィールドを表示します。

関連項目

- カタログに関連付けられたワークフローの入力フィールドの取得, （109 ページ）
- ワークフローの入力フィールドの取得, （113 ページ）
- ワークフローの出力フィールドの取得, （116 ページ）

ワークフローの入力フィールドの取得

目標

ワークフローの入力フィールドを取得します。ワークフローの入力ラベル、名前、説明、タイプ、および isAdmin 入力タイプなど、ワークフローの入力フィールドを表示できます。

コンテキスト

ワークフロー入力フィールドを確認します。

前提条件

ワークフロー入力が Cisco UCS Director に存在している必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIGetWorkflowInputs&opData=
{param0:"Expand VSAN Cluster"}
```

応答


```
{ "serviceResult":{"details":[{"name":"input_6_VSAN_Cluster915","label":"VSAN Cluster",
"description":"Select VSAN Cluster","type":"vsanCluster","catalogType":null,"isOptional":false,
"inputFieldType":"popup-table","isMultiSelect":false,"inputFieldValidator":null,
"isAdminInput":false},{"name":"input_3_Host_Nodes50","label":"Host Nodes","description":"Host

 Nodes : Ex. 172.29.195.75,172.29.195.76,172.29.195.77","type":"gen_text_input",
"catalogType":null,"isOptional":false,"inputFieldType":"text","isMultiSelect":false,
"inputFieldValidator":null,"isAdminInput":false},{"name":"input_2_Host_User_ID924",
"label":"Host User ID","description":"Enter User ID","type":"gen_text_input","catalogType":null,
"isOptional":false,"inputFieldType":"text","isMultiSelect":false,"inputFieldValidator":null,
"isAdminInput":false},{"name":"input_3_Host_Password766","label":"Host Password","description":
"Enter Host Password","type":"password","catalogType":null,"isOptional":false,
"inputFieldType":"password","isMultiSelect":false,"inputFieldValidator":null,
"isAdminInput":false},{"name":"input_6_Host_License325","label":"Host License","description":"",
"type":"gen_text_input","catalogType":null,"isOptional":true,"inputFieldType":"text",
"isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_5_DVSwitch176","label":"DVSwitch","description":"Select DVSwitch","type":
"VMwareDVSwitchIdentity","catalogType":null,"isOptional":false,"inputFieldType":"table",
"isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_9_DVSwitch_Uplink_Portgroup885","label":"DVSwitch Uplink Portgroup",
"description":"Select Uplink Portgroup,According to DVSwitch","type":"uplinkPortGroupLovList",
"catalogType":null,"isOptional":false,"inputFieldType":"embedded-lov",
"isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_10_Virtual_SAN_Portgroup153","label":"Virtual SAN Portgroup",
"description":"Select Virtual SAN Portgroup","type":"VMwareDVPortgroupIdentity",
"catalogType":null,"isOptional":false,"inputFieldType":"table","isMultiSelect":false,
"inputFieldValidator":null,"isAdminInput":false},{"name":"input_11_VSAN_IP_Pool_Policy298",
"label":"VSAN IP Pool Policy","description":"Select VSAN IP Pool Policy","type":
"VMWareIPPoolPolicy","catalogType":null,"isOptional":false,"inputFieldType":"popup-table",
"isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_11_vMotion_Portgroup289","label":"vMotion Portgroup",
"description":"","type":"VMwareDVPortgroupIdentity","catalogType":null,"isOptional":true,
"inputFieldType":"table","isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_10_vMotion_IP_Pool_Policy807","label":"vMotion IP Pool Policy","description":"",
"type":"VMWareIPPoolPolicy","catalogType":null,"isOptional":true,"inputFieldType":"popup-table",
"isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false},
{"name":"input_7_MTU_Size697","label":"MTU Size","description":"Enter MTU Size,
Ex : 1500 or 9000","type":"gen_text_input","catalogType":null,"isOptional":false,
"inputFieldType":"text","isMultiSelect":false,"inputFieldValidator":null,"isAdminInput":false}]},

 "serviceError":null, "serviceName":"InfraMgr","opName":"userAPIGetWorkflowInputs" }
```


コンポーネント

param0：ワークフローの名前。

コード

```
public class GetWorkflowInputs
{
 public static void main(String[] args) throws Exception
{
  CuicServer api = CuicServer.getAPI("10.23.210.42", "3E17CFFBA7A64C71B8958F40DE2EC9B3",
 "http", 80);
  UserAPIGlobal instance = new UserAPIGlobal(api);
       APIWorkflowInputDetails details = instance.userAPIGetWorkflowInputs("Test1");
       List<APIWorkflowInputDetail> list = null;
    if(details != null){
     list = details.getDetails();
    }else{
     throw new Exception("No workflow input defination found!");
    }
       if(list != null && list.size() > 0){
        for(int i = 0;i < list.size();i++){
        System.out.println("Field Name: "+list.get(i).getName());
        System.out.println("Field Label: "+list.get(i).getLabel());
        System.out.println("Field Description: "+list.get(i).getDescription());
        System.out.println("Input Type: "+list.get(i).getType());
        System.out.println("Field Type: "+list.get(i).getInputFieldType());
        System.out.println("Is Admin Input Type :"+list.get(i).isAdminInput());
       }
       }else{
        System.out.println("No workflow input found!");
       }
 }
}
```

結果

ワークフローの入力フィールドがリストされます。

結果サンプル：

- Field Name: input_0_name250
- Field Label: name
- Field Description: person name
- Input Type: gen_text_input
- Field Type: text
- Is Admin Input Type : false

実装

userAPIGetWorkflowInputs API を使用して、ワークフローの名前を渡し、ワークフローの入力フィールドを表示します。

関連項目

- カタログに関連付けられたワークフローの入力フィールドの取得, （109 ページ）
- カタログに関連付けられたワークフローの出力フィールドの取得, （111 ページ）
- ワークフローの出力フィールドの取得, （116 ページ）

ワークフローの出力フィールドの取得

目標

ワークフローの出力フィールドを取得します。出力ラベル、名前、説明、およびワークフローのタイプなどの出力フィールドを取得できます。

コンテキスト

ワークフローの出力フィールドを確認します。

前提条件

なし

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
workflow:userAPIGetWorkflowOutputDefinition&opData={param0:"Test1"}
```

応答

```
{ "serviceResult":{"workflowOutputFieldList":[{"outputFieldLabel":"OUTPUT_NAME",
"outputFieldName":"output_0_OUTPUT_NAME75","outputFieldType":"gen_text_input",
"outputFieldDescription":"person's name"}]}, "serviceError":null,
"serviceName":"InfraMgr",
 "opName":"workflow:userAPIGetWorkflowOutputDefinition" }
```

コンポーネント

param0：出力フィールドを表示させるワークフローの名前。

コード

```
public class GetWorkflowOutputDef
{
 public static void main(String[] args) throws Exception
 {
  CuicServer api = CuicServer.getAPI("10.23.210.42", "3E17CFFBA7A64C71B8958F40DE2EC9B3",
 "http", 80);
  UserAPIWorkflow instance = new UserAPIWorkflow(api);
  APIWorkflowOutputFieldDefinitionList def =
instance.userAPIGetWorkflowOutputDefinition("Test1");
  List<APIWorkflowOutputFieldDefinition> list =null;
  if(def != null){
   list = def.getWorkflowOutputFieldList();
  }else{
   throw new Exception("No workflow output defination found!");
  }
  if(list != null && list.size() > 0){
   for(int i=0;i<list.size();i++){
    System.out.println("Field Name: "+list.get(i).getOutputFieldName());
    System.out.println("Field Label: "+list.get(i).getOutputFieldLabel());
   System.out.println("Field Description: "+list.get(i).getOutputFieldDescription());

    System.out.println("Field Type: "+list.get(i).getOutputFieldType());
   }
  }else{
   System.out.println("No workflow output field found!");
  }
 }
}
```

結果

ワークフローの出力フィールドがリストされます。

結果サンプル：

- Field Name: output_0_OUTPUT_NAME75
- Field Label: OUTPUT_NAME
- Field Description: person's name
- Field Type: gen_text_input

実装

userAPIGetWorkflowOutputDefinition API を使用して、ワークフローの名前を渡し、ワークフローの出力フィールドを表示します。

関連項目

- カタログに関連付けられたワークフローの入力フィールドの取得, （109 ページ）
- カタログに関連付けられたワークフローの出力フィールドの取得, （111 ページ）
- ワークフローの入力フィールドの取得, （113 ページ）

MSP の管理

MSP モードの切り替え

目標

Cisco UCS Director のマネージド サービス プロバイダー（MSP）モードを有効化または無効化します。

コンテキスト

Cisco UCS Director で、MSP ユーザまたは MSP 組織へのアクセスに MSP モードが有効にされている必要があります。

前提条件

なし

実行後の必須作業

Cisco UCS Director サービスを再起動します。

REST の URL

```
/app/api/rest?formatType=json&opName=auth:userAPIToggleMspMode&opData={param0:
{"action":true,"tenantName":"sample","orgName":"sample"}}
```

コンポーネント

userAPIToggleMspMode API のパラメータは次のとおりです。

- action：MSP モードを有効化するには、値を true に設定します。MSP モードを無効化するには、false に設定します。
- tenantName：MSP ユーザの名前。MSP 無効化操作の場合、このパラメータはオプションです。
- orgName：MSP 組織の名前MSP 無効化操作の場合、このパラメータはオプションです。

コード

```
public static void main(String[] args) throws Exception
{
  CuicServer api = CuicServer.getAPI("172.29.110.241",
"3E17CFFBA7A64C71B8958F40DE2EC9B3", "http", 80);
  UserAPIAuthConfig instance = new UserAPIAuthConfig(api);
  APIMSPModeParams params = new APIMSPModeParams();
  params.setAction(true);
  params.setTenantName("MSP Users");
  params.setOrgName("MSP Organization");
  String result = instance.userAPIToggleMspMode(params);
  System.out.println("Result: "+result);
 }
```

結果

設定された MSP モードに応じてステータス メッセージが表示されます。

- MSP モードが有効化されると、次のメッセージが表示されます。
  ```
  MSP Mode is enabled successfully. Restart the Cisco UCS Director services to
  reflect the changes.
  ```
- MSP モードが無効化されると、次のメッセージが表示されます。
  ```
  MSP Mode is disabled successfully. Restart the Cisco UCS Director services to
  reflect the changes.
  ```

実装

MSP ユーザ名および MSP 組織名を渡し、userAPIToggleMspMode API で action を true に設定して、MSP モードを有効にします。

データストアの管理

適格なデータストア クラスタのリストの取得

目標

既存の VM に新しいディスクを追加するために使用できる適格なデータストア クラスタのリストを取得します。

コンテキスト

既存の VM に新しいディスクを追加する際は、有効かつ使用可能なデータストア クラスタを用意する必要があります。userAPIGetEligibleDataStoreClustersForCreateNewDisk API を使用して、新しいディスクを追加する際に有効なデータストア クラスタを用意するために使用できる適格なデータストア クラスタのリストを取得できます。この API は、vmId、diskSize、および diskType パラメータに基づく適格なデータストア クラスタのリストを返します。

前提条件

vmId は Cisco UCS Director にすでに存在する VM を参照する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=genericvm:
userAPIGetEligibleDataStoreClustersForCreateNewDisk&opData={param0:{"vmId":86,
"diskSize":10.0,"diskType":0}}
```

応答

```
{"serviceResult":{"name":"DS_Cluster_2","capacity":1099511627776,"freespace":26096259072,
"drsEnabled":"Enabled","ioMetrics":"Enabled","automationLevel":"automated","utilitySpaceThreshold":50
"ioLatencyThreshold":5,"datastores":"QA_DS01","type":"NFS","accountName":"VMC001",
"vmwareDatacenterName":"New Datacenter"}], "serviceError":null, "serviceName":"InfraMgr",
"opName":"genericvm:userAPIGetEligibleDataStoreClustersForCreateNewDisk" }
```

コンポーネント

- int vmId：Cisco UCS Director の既存の VM の VM ID。
- double diskSize：追加する必要があるディスクのサイズ（GB 単位）。
- int diskType：追加する必要があるディスクのタイプ。有効なディスク タイプは次のとおりです。
  - 0：システム
  - 1：データ
  - 2：データベース
  - 3：ログ
  - 4：スワップ

コード

```
import java.util.List;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIVMware;
import com.cisco.cuic.api.models.vmware.CreateVMDiskParams;
import com.cisco.cuic.api.models.vmware.VMWareDatastoreCluster;

public class TestUserAPIGetEligibleDataStoreClustersForCreateNewDisk {
 public static void main(String[] args)  {
  CuicServer server = CuicServer.getAPI("172.22.234.172",
"91E7A134048F45FD88A29FCBBFD2C233", "https", 443, 12000, "none", "");
  UserAPIVMware instance = new UserAPIVMware(server);
  CreateVMDiskParams params = new CreateVMDiskParams();
  params.setVmId(86);
  params.setDiskSize(10.0);
  params.setDiskType(0);
  List<VMWareDatastoreCluster> vmwareDatastoreClusterList;
  try {
   vmwareDatastoreClusterList =
instance.userAPIGetEligibleDataStoreClustersForCreateNewDisk(params);
   System.out.println("list size = "+vmwareDatastoreClusterList.size());
   for(VMWareDatastoreCluster vmwareDatastoreCluster:vmwareDatastoreClusterList){
    System.out.println(vmwareDatastoreCluster.getName());
   }
  } catch (Exception e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
  }
 }
}
```

結果

渡した vmId、diskSize、および diskType パラメータに基づくデータストア クラスタのリストが返されます。

実装

コンポーネント セクションに記載されているパラメータを設定して、使用可能なデータストア クラスタのリストを取得します。

関連項目

適格なデータ ストアのリストの取得

適格なデータ ストアのリストの取得

目標

既存の VM に新しいディスクを追加するために使用できる適格なデータストアのリストを取得します。

コンテキスト

既存の VM に新しいディスクを追加する際は、有効かつ使用可能なデータストアを用意する必要があります。userAPIGetEligibleDataStoresForCreateNewDisk API を使用して、新しいディスクを追加する際に有効なデータストアを用意するために使用できる適格なデータストアのリストを取得できます。この API は、vmId、diskSize、および diskType パラメータに基づく適格なデータストアのリストを返します。

前提条件

vmId は Cisco UCS Director にすでに存在する VM を参照する必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=
genericvm:userAPIGetEligibleDataStoresForCreateNewDisk&opData={param0:{"vmId":86,
"diskSize":20.0,"diskType":0}}
```

応答


```
{ "serviceResult":[{"hostName":"172.29.109.65","accountName":"VMC001","name":
"datastore1 (2)","capacityBytes":993211187200,"freeBytes":37044092928,"type":"VMFS",
"url":"ds:///vmfs/volumes/51b73895-bfedae18-1852-4c4e353d287e/","isMultiHost":false,
"uncomitted":515370287104,"vmCount":26,"numHosts":1,"mountAccessMode":"readWrite",
"mountPath":"/vmfs/volumes/51b73895-bfedae18-1852-4c4e353d287e","vmfsBlockSizeMB":1,
"vmfsMaxBlocks":63963136,"vmfsVersion":"5.58","luns":[{"uuid":null,"canonicalName":
"naa.600605b005a7ee201949f39808290917","lunType":"disk","scsiLevel":5,"vendor":"LSI",
"key":"key-vim.host.ScsiDisk-0200000000600605b005a7ee201949f398082909174d5239323731",
"deviceType":"disk","displayName":"Local LSI Disk (naa.600605b005a7ee201949f39808290917)",
"model":"MR9271-8i","deviceName":"/vmfs/devices/disks/naa.600605b005a7ee201949f39808290917",
"hostName":"10.10.109.65","accountName":"VMC001","datacenterName":"New Datacenter",
"datastoreName":"datastore1 (2)"}],"nfsRemoteHost":null,"nfsRemotePath":null,
"nfsRemoteUsername":null,"vmwareDatacenterName":"New Datacenter","adapterLunMap":[],
"vmList":null,"accessible":true,"localfsRemotePath":null}], "serviceError":null,
"serviceName":"InfraMgr",
"opName":"genericvm:userAPIGetEligibleDataStoresForCreateNewDisk" }
```


コンポーネント

- int vmId：Cisco UCS Director の既存の VM の VM ID。
- double diskSize：追加する必要があるディスクのサイズ（GB 単位）。
- int diskType：追加する必要があるディスクのタイプ。有効なディスク タイプは次のとおりです。
  - 0：システム
  - 1：データ
  - 2：データベース
  - 3：ログ
  - 4：スワップ

コード

```
import java.util.List;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIVMware;
import com.cisco.cuic.api.models.vmware.CreateVMDiskParams;
import com.cisco.cuic.api.models.vmware.DataStoreInfo;

public class TestUserAPIGetEligibleDataStoresForCreateNewDisk {
 public static void main(String[] args) {
  CuicServer server = CuicServer.getAPI("172.22.234.172",
"91E7A134048F45FD88A29FCBBFD2C233", "https", 443, 12000, "none", "");
  UserAPIVMware instance = new UserAPIVMware(server);
  CreateVMDiskParams params = new CreateVMDiskParams();
  params.setVmId(86);
  params.setDiskSize(10.0);
  params.setDiskType(0);
  List<DataStoreInfo> dataStoreInfoList;
  try {
   dataStoreInfoList = instance.userAPIGetEligibleDataStoresForCreateNewDisk(params);

   for(DataStoreInfo datastoreInfo: dataStoreInfoList){
    System.out.println("Data store name: "+datastoreInfo.getName());
   }
  } catch (Exception e) {

   e.printStackTrace();
  }
 }
}
```

結果

渡した vmId、diskSize、および diskType パラメータに基づくデータストアのリストが返されます。

実装

コンポーネント セクションに記載されているパラメータを設定して、使用可能なデータストアのリストを取得します。

関連項目

適格なデータストア クラスタのリストの取得

レポートの管理

使用可能なレポート定義の表示

目標

userAPIGetAvailableReports API を使用して、指定した contextName と contextValue に対して Cisco UCS Director で利用可能なすべてのレポートのタイプ、ID およびラベルを表示できます。

コンテキスト

指定した contextName と contextValue に対して利用可能なレポートの定義を表示します。

前提条件

指定する contextName と contextValue が Cisco UCS Director で使用可能であることが必要です。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIGetAvailableReports&opData
={param0:"22",param1:"Jha_VDC_VMware_82"}
```

応答


```
{ "serviceResult":[{"reportLabel":"Summary","reportId":"SUMMARY-V0","reportType":"tabular"},
{"reportLabel":"Summary","reportId":"SUMMARY-V14","reportType":"tabular"},{"reportLabel":"VMs",
"reportId":"VMS-T0","reportType":"tabular"},{"reportLabel":"Events","reportId":"EVENTS-T0",
"reportType":"tabular"},{"reportLabel":"Static IP Assignment",
"reportId":"STATIC-IP-ASSIGNMENT-T0","reportType":"tabular"},{"reportLabel":"Deleted VMs",
"reportId":"DELETED-VMS-T0","reportType":"tabular"},{"reportLabel":"VDC Compliance on VMs",
"reportId":"VDC-COMPLIANCE-ON-VMS-T0","reportType":"tabular"},{"reportLabel":"Trend:vDC CPU
 Usage",
"reportId":"TREND-VDC-CPU-USAGE-H0","reportType":"trend"},{"reportLabel":"Trend:vDC Memory
 Usage",
"reportId":"TREND-VDC-MEMORY-USAGE-H0","reportType":"trend"},{"reportLabel":"Trend:vDC Disk
 Usage",
"reportId":"TREND-VDC-DISK-USAGE-H0","reportType":"trend"},{"reportLabel":"Trend:vDC Network
 Usage",
"reportId":"TREND-VDC-NETWORK-USAGE-H0","reportType":"trend"},{"reportLabel":"Trend:Number
 of VMs",
"reportId":"TREND-NUMBER-OF-VMS-H0","reportType":"trend"},{"reportLabel":"Trend:VM Additions

& Deletions","reportId":"TREND-VM-ADDITIONS-&-DELETIONS-H0","reportType":"trend"},
{"reportLabel":"Trend: vDC CPU Usage (GHz)","reportId":"TREND-VDC-CPU-USAGE-(GHZ)-H14",
"reportType":"trend"},{"reportLabel":"Trend: vDC Memory Usage GB","reportId":
"TREND-VDC-MEMORY-USAGE-GB-H14","reportType":"trend"},{"reportLabel":"Trend: vDC Disk Usage",
"reportId":"TREND-VDC-DISK-USAGE-H14","reportType":"trend"},{"reportLabel":"Trend: vDC
Network Usage",
"reportId":"TREND-VDC-NETWORK-USAGE-H14","reportType":"trend"},
{"reportLabel":"Trend: Number of
VMs","reportId":"TREND-NUMBER-OF-VMS-H14","reportType":"trend"},
{"reportLabel":"Trend: VM Additions &
Deletions","reportId":"TREND-VM-ADDITIONS-&-DELETIONS-H14",
"reportType":"trend"},{"reportLabel":"Trend: Total Cost","reportId":"TREND-TOTAL-COST-H0",
"reportType":"trend"},{"reportLabel":"Trend: VM Cost","reportId":"TREND-VM-COST-H0",
"reportType":"trend"},{"reportLabel":"Trend: CPU Cost","reportId":"TREND-CPU-COST-H0",
"reportType":"trend"},{"reportLabel":"Trend: Memory Cost","reportId":"TREND-MEMORY-COST-H0",
"reportType":"trend"},{"reportLabel":"Trend: Network Cost","reportId":"TREND-NETWORK-COST-H0",
"reportType":"trend"},{"reportLabel":"Trend: Total Cost","reportId":"TREND-TOTAL-COST-H14",
"reportType":"trend"},{"reportLabel":"Trend: VM Cost","reportId":"TREND-VM-COST-H14",
"reportType":"trend"},{"reportLabel":"Trend: CPU Cost","reportId":"TREND-CPU-COST-H14",
"reportType":"trend"},{"reportLabel":"Trend: Memory Cost","reportId":"TREND-MEMORY-COST-H14",
"reportType":"trend"},{"reportLabel":"Trend: Network
Cost","reportId":"TREND-NETWORK-COST-H14",
"reportType":"trend"},{"reportLabel":"Trend:Snapshot File
Size","reportId":"TREND-SNAPSHOT-FILE-SIZE-H0",
"reportType":"trend"},{"reportLabel":"VMs","reportId":"VMS-T14","reportType":"tabular"},
{"reportLabel":"Events","reportId":"EVENTS-T14","reportType":"tabular"},{"reportLabel":"Deleted
 VMs",
"reportId":"DELETED-VMS-T14","reportType":"tabular"},{"reportLabel":"Chargeback",
"reportId":"CHARGEBACK-T12","reportType":"tabular"},{"reportLabel":"Resource Accounting",
"reportId":"RESOURCE-ACCOUNTING-T12","reportType":"tabular"},{"reportLabel":"Resource
Accounting Details",
"reportId":"RESOURCE-ACCOUNTING-DETAILS-T12","reportType":"tabular"},
{"reportLabel":"CPU Usage (MHz)","reportId":"CPU-USAGE-(MHZ)-S0","reportType":"snapshot"},
{"reportLabel":"Memory Usage (GB)","reportId":"MEMORY-USAGE-(GB)-S0","reportType":"snapshot"},
{"reportLabel":"Disk Usage (GB)","reportId":"DISK-USAGE-(GB)-S0","reportType":"snapshot"},
{"reportLabel":"Number of Events by Severity","reportId":"NUMBER-OF-EVENTS-BY-SEVERITY-S0",
"reportType":"snapshot"},{"reportLabel":"CPU Usage (MHz)","reportId":"CPU-USAGE-(MHZ)-S14",
"reportType":"snapshot"},{"reportLabel":"Memory Usage
(GB)","reportId":"MEMORY-USAGE-(GB)-S14",
"reportType":"snapshot"},{"reportLabel":"Disk Usage (GB)","reportId":"DISK-USAGE-(GB)-S14",
"reportType":"snapshot"}], "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPIGetAvailableReports" }
```

コンポーネント

userAPIGetAvailableReports API のパラメータは次のとおりです。

- String param0：レポート コンテキストの名前
- int param1：レポート コンテキストの値。

レポート コンテキストの名前と値の詳細については、『Cisco UCS Director Open Automation Cookbook』の付録 B：「Report Context Types and Report Context Names」を参照してください。

コード

```
import java.util.List;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.client.APIReportDefinition;
import com.cisco.cuic.api.client.CuicServer;

public class TestuserAPIGetAvailableReports {

 public static void main(String[] args) throws Exception {
   CuicServer server =
CuicServer.getAPI("172.29.110.222","96408900345D40C0BC889E4F41C2E094", "https", 443);

   UserAPIGlobal instance = new UserAPIGlobal(server);
   List<APIReportDefinition>
apiReportDefinitionList=instance.userAPIGetAvailableReports("22", "Jha_VDC_VMware-82");

   int i=0;
   for(APIReportDefinition apiReportDefinition: apiReportDefinitionList){
    System.out.println("Report_"+i);
    System.out.println("reportLabel: "+apiReportDefinition.getReportLabel());
    System.out.println("reportID: "+apiReportDefinition.getReportId());
    System.out.println("reportType: "+apiReportDefinition.getReportType());
    i++;
   }
 }
}
```

結果

指定した contextName と contextValue に対して利用可能なレポートのタイプ、ID、およびラベルが表示されます。

実装

userAPIGetAvailableReports API を使用して、contextName と contextValue を渡し、指定した contextName と contextValue の API レポート定義のリストを表示します。

関連項目

履歴レポートの表示

リソース使用レポートの表示

スナップショット レポートの表示

表形式レポートの表示

履歴レポートの表示

目標

userAPIGetHistoricalReport API を使用して、特定の期間についての VDC CPU 使用傾向レポートなど、あるレポート コンテキストの履歴データを表示できます。レポート コンテキストでは、contextName、contextValue、および reportId が参照されます。

コンテキスト

特定の期間のレポート コンテキストの履歴データが表示されます。

前提条件

指定する contextName と contextValue が Cisco UCS Director で使用可能であることが必要です。期間は、Cisco UCS Director 内の指定された期間のいずれか、または Cisco UCS Director でサポートされているフォーマットに従ってカスタマイズされた時間帯であることが必要です。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIGetHistoricalReport&opData=
{param0:"22",param1:"2",param2:"TREND-VDC-CPU-USAGE-H0",param3:"hourly"}
```

応答

```
{ "serviceResult":{"series":[{"paramName":"vdc.cpu.usage.average:gigaHertz",
"paramLabel":"CPU Usage (GHz)","values":[{"timestamp":1467594752222,"min":0.0,"max":0.0,
"avg":0.165},{"timestamp":1467598352222,"min":0.0,"max":0.0,"avg":0.165}],"precision":2,"units":""

 "serviceError":null, "serviceName":"InfraMgr", "opName":"userAPIGetHistoricalReport" }
```

コンポーネント

userAPIGetHistoricalReport API のパラメータは次のとおりです。

- String param0：レポート コンテキストの名前
- int param1：レポート コンテキストの値。
- int param2：レポートの固有識別子。
- int Param3：履歴レポートを生成する対象となる期間。事前定義された期間の有効な値は時間単位、日単位、週単位、月単位です。カスタム期間は、プレフィクス「custom:」で始まる必要があります。

レポート コンテキストの名前と値の詳細については、『Cisco UCS Director Open Automation Cookbook』の付録 B：「Report Context Types and Report Context Names」を参照してください。

コード

```
import java.util.List;
import com.cisco.cuic.api.client.APIHistoricalReport;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.client.DataSample;
import com.cisco.cuic.api.client.HistoricalDataSeries;
import com.cisco.cuic.api.models.UserAPIGlobal;

public class TestuserAPIGetHistoricalReport {

 public static void main(String[] args) throws Exception {
  CuicServer server =
CuicServer.getAPI("172.29.110.222","96408900345D40C0BC889E4F41C2E094", "https", 443);

  UserAPIGlobal instance = new UserAPIGlobal(server);
  APIHistoricalReport historicalReport=instance.userAPIGetHistoricalReport("22", "2",
 "TREND-VDC-CPU-USAGE-H0", "hourly");
  List<HistoricalDataSeries> historicalDataSeriesList = historicalReport.getSeries();

  int i=0;
  for(HistoricalDataSeries hds: historicalDataSeriesList){
   System.out.println("****Series_"+i+"****");
   System.out.println("paramName: "+hds.getParamName());
   System.out.println("paramLabel: "+hds.getParamLabel());
   DataSample[] dataSampleArr=hds.getValues();
   for(DataSample ds:dataSampleArr ){
    System.out.println("timestamp: "+ds.getTimestamp());
    System.out.println("min: "+ds.getMin());
    System.out.println("max: "+ds.getMax());
    System.out.println("avg: "+ ds.getAvg());
   }
   System.out.println("precision: "+hds.getPrecision());
   System.out.println("units: "+hds.getUnits());
   i++;
  }
 }
}
```

結果

特定の期間のレポートのコンテキストの履歴データが表示されます。

実装

userAPIGetHistoricalReport API を使用して、contextName、contextValue、および期間を渡し、特定の期間中のレポート コンテキストの履歴データを表示します。

関連項目

使用可能なレポート定義の表示

リソース使用レポートの表示

スナップショット レポートの表示

表形式レポートの表示

リソース使用レポートの表示

目標

クラウドインフラストラクチャ管理に使用されるリソースのレポートは、userAPIGetResourceUsageCostSummary API を使用して表示できます。この API を使用して、管理者とエンドユーザが、日単位、週単位、月単位のリソース使用レポートを確認できます。

コンテキスト

日単位、週単位、月単位のリソースのインフラストラクチャ使用率やコスト レポートを表示します。

前提条件

なし

REST の URL

このセクションには、さまざまなシナリオでリソース使用レポートを表示する REST URL が準備されています。

例 1：resourceName のない仮想マシン（VM）の日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:{"requestParam":
[{"name":"vmid","value":"56"}],"fromTimeInMilliSeconds":1442946600000,
"toTimeInMilliSeconds":1443032999000}}
```

応答

```
{ "serviceResult":{"resourceType":"VM","duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"vmid","value":"56"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,
"applicationCost":23.0,"totalCost":5.0,"fixedCost":4.0,"catalogItemName":
"SDK","cpuResourceUsageCost":{"cpu_GHz_Hours":111.99998388000004,
"cpuCost":4.0,"reservedCpuGhzCost":9.0,"usedCpuGhzCost":65.0,"reservedCpuGhz":0.0,
"usedCpuGhz":0.0,"cpuCores":40,"cpuCoreCost":56.0}],"diskResourceUsageCost":
[{"committedDiskGB":40.60000000000001,"uncommittedDiskGB":1.4000000000000008,
"committedDiskGBCost":56.0,"uncommittedDiskGBCost":567.0}],"memoryResourceUsageCost":
[{"memory":67.0,"memoryCost":67.0,"reservedMemoryGBCost":7.0,"usedMemoryGBCost":12.0,
"usedMemoryGB":0.0,"reservedMemoryGB":0.0}],"networkResourceUsageCost":
[{"netRxUsageGB":0.0,"netTxUsageGB":76.0,"netRxUsageGBCost":0.00006389617919921875,
"netTxUsageGBCost":0.0000829696655273437 5}],"physicalServerResourceUsageCost":
[{"halfLengthBlade":false,"bladeType":"","fullBladeCost":67.0,"halfBladeCost":76.0,
"serverCost":5.0}]}]}, "serviceError":null, "serviceName":"InfraMgr",
"opName":"chargeback:userAPIGetResourceUsageCostSummary" }
```

例 2：resourceName のないグループの日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:{"requestParam":
[{"name":"Group","value":"Default Group"}],"fromTimeInMilliSeconds":1442946600000,
"toTimeInMilliSeconds":1443032999000}}
```

応答

```
{ "serviceResult":{"resourceType":null,"duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"Group","value":"Default Group"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,"applicationCost":23.0,
"totalCost":5.0,"fixedCost":4.0,"catalogItemName":"SDK","cpuResourceUsageCost":
[{"cpu_GHz_Hours":351.99997083999995,"cpuCost":4.0,"reservedCpuGhzCost":9.0,
"usedCpuGhzCost":65.0,"reservedCpuGhz":0.0,"usedCpuGhz":0.0,"cpuCores":120,
"cpuCoreCost":56.0}],"diskResourceUsageCost":[{"committedDiskGB":1225.1200000000001,
"uncommittedDiskGB":3974.999999999999,"committedDiskGBCost":56.0,
"uncommittedDiskGBCost":567.0}],"memoryResourceUsageCost":[{"memory":240.0,
"memoryCost":67.0,"reservedMemoryGBCost":7.0,"usedMemoryGBCost":12.0,
"usedMemoryGB":0.0,"reservedMemoryGB":0.0}],"networkResourceUsageCost":
[{"netRxUsageGB":0.0,"netTxUsageGB":76.0,"netRxUsageGBCost":0.00006389617919921875,
"netTxUsageGBCost":0.0000829696655273437 5}],"physicalServerResourceUsageCost":
[{"halfLengthBlade":false,"bladeType":"","fullBladeCost":67.0,"halfBladeCost":76.0,
"serverCost":5.0}]}]}, "serviceError":null, "serviceName":"InfraMgr",
"opName":"chargeback:userAPIGetResourceUsageCostSummary" }
```

例 3：resourceName のない仮想データセンター（VDC）の日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:
{"requestParam":[{"name":"vdcid","value":"1"}],"fromTimeInMilliSeconds":1442946600000,
"toTimeInMilliSeconds":1443032999000}}
```

応答

```
{ "serviceResult":{"resourceType":null,"duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"vdcName","value":"Default vDC"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,"applicationCost":23.0,
"totalCost":5.0,"fixedCost":4.0,"catalogItemName":"SDK","cpuResourceUsageCost":
```

[{"cpu_GHz_Hours":351.99997083999995,"cpuCost":4.0,"reservedCpuGhzCost":9.0,
"usedCpuGhzCost":65.0,"reservedCpuGhz":0.0,"usedCpuGhz":0.0,"cpuCores":120,
"cpuCoreCost":56.0}],"diskResourceUsageCost":[{"committedDiskGB":1225.1200000000001,
"uncommittedDiskGB":3974.999999999999,"committedDiskGBCost":56.0,
"uncommittedDiskGBCost":567.0}],"memoryResourceUsageCost":[{"memory":240.0,
"memoryCost":67.0,"reservedMemoryGBCost":7.0,"usedMemoryGBCost":12.0,
"usedMemoryGB":0.0,"reservedMemoryGB":0.0}],"networkResourceUsageCost":
[{"netRxUsageGB":0.0,"netTxUsageGB":76.0,"netRxUsageGBCost":0.00006389617919921875,
"netTxUsageGBCost":0.00008296966552734375}],"physicalServerResourceUsageCost":
[{"halfLengthBlade":false,"bladeType":"","fullBladeCost":67.0,"halfBladeCost":76.0,
"serverCost":5.0}]}]}, "serviceError":null, "serviceName":"InfraMgr",
"opName":"chargeback:userAPIGetResourceUsageCostSummary" }

例 4：VM 内の CPU の日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:{"requestParam":
[{"name":"vmid","value":"56"},{"name":"resourceName","value":"CPU"}],
"fromTimeInMilliSeconds":1442946600000,"toTimeInMilliSeconds":1443032999000}}
```

応答

{ "serviceResult":{"resourceType":"VM","duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"vmid","value":"56"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,
"applicationCost":23.0,"totalCost":5.0,"fixedCost":4.0,"catalogItemName":
"SDK","cpuResourceUsageCost":{"cpu_GHz_Hours":111.99998388000004,"cpuCost":4.0,
"reservedCpuGhzCost":9.0,"usedCpuGhzCost":2.6,"reservedCpuGhz":0.0,"usedCpuGhz":0.0,
"cpuCores":40,"cpuCoreCost":56.0}],"diskResourceUsageCost":null,
"memoryResourceUsageCost":null,"networkResourceUsageCost":null,
"physicalServerResourceUsageCost":null}]}, "serviceError":null,
"serviceName":"InfraMgr", "opName":"chargeback:userAPIGetResourceUsageCostSummary" }

例 5：グループ内のディスクの日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:
{"requestParam":[{"name":"Group","value":"Default Group"},
{"name":"resourceName","value":"disk"}],"fromTimeInMilliSeconds":1442946600000,
"toTimeInMilliSeconds":1443032999000}}
```

応答

{ "serviceResult":{"resourceType":null,"duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"Group","value":"Default Group"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,"applicationCost":23.0,
"totalCost":5.0,"fixedCost":4.0,"catalogItemName":"SDK","cpuResourceUsageCost":null,
"diskResourceUsageCost":[{"committedDiskGB":1225.1200000000001,
"uncommittedDiskGB":3974.999999999999,"committedDiskGBCost":56.0,
"uncommittedDiskGBCost":567.0}],"memoryResourceUsageCost":null,
"networkResourceUsageCost":null,"physicalServerResourceUsageCost":null}]},
 "serviceError":null, "serviceName":"InfraMgr",
"opName":"chargeback:userAPIGetResourceUsageCostSummary" }

例 6：VDC 内の VM の日単位使用レポート

要求

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:
{"requestParam":[{"name":"vdcid","value":"1"},{"name":"resourceName",
"value":"VM"}],"fromTimeInMilliSeconds":1442946600000,
"toTimeInMilliSeconds":1443032999000}}
```

応答

```
{ "serviceResult":{"resourceType":null,"duration":"Daily_22_9_2015",
"responseParamList":{"list":[{"name":"vdcName","value":"Default vDC"}]},
"resourceUsageCostSummary":[{"groupId":1,"groupName":"Default Group","vdcId":0,
"onetimeCost":8.0,"activeCost":56.0,"inactiveCost":4.0,"applicationCost":23.0,
"totalCost":5.0,"fixedCost":4.0,"catalogItemName":"SDK","cpuResourceUsageCost":
[{"cpu_GHz_Hours":351.99997083999995,"cpuCost":4.0,"reservedCpuGhzCost":9.0,
"usedCpuGhzCost":65.0,"reservedCpuGhz":0.0,"usedCpuGhz":0.0,"cpuCores":0,
"cpuCoreCost":56.0}],"diskResourceUsageCost":[{"committedDiskGB":1225.1200000000001,
"uncommittedDiskGB":3974.999999999999,"committedDiskGBCost":56.0,
"uncommittedDiskGBCost":567.0}],"memoryResourceUsageCost":[{"memory":240.0,
"memoryCost":67.0,"reservedMemoryGBCost":7.0,"usedMemoryGBCost":12.0,
"usedMemoryGB":0.0,"reservedMemoryGB":0.0}],"networkResourceUsageCost":
[{"netRxUsageGB":0.0,"netTxUsageGB":76.0,"netRxUsageGBCost":0.00006389617919921875,
"netTxUsageGBCost":0.00008296966552734375}],"physicalServerResourceUsageCost":null}]},

 "serviceError":null, "serviceName":"InfraMgr",
"opName":"chargeback:userAPIGetResourceUsageCostSummary" }
```

コンポーネント

userAPIGetResourceUsageCostSummary API のパラメータは次のとおりです。

- APINameValueList requestParam：リクエストされたレポートパラメータの、名前と値のペア形式のリスト。名前の値は vmId、vdcName、groupName、および resourceName です。名前として resourceName が渡された場合、resourceName の値は cpu、disk、memory、VM、BM、network、または空の値（" "）になります。

  リソース使用レポートを表示するパラメータの組み合わせ：

名前	resourceName
vmid	CPU /Disk/VM/BM/Memory/Network
vdcid	Disk
Group	VM
vmid	なし
vdcid	なし
Group	なし

- fromTimeInMilliSeconds：レポートの開始時間（ミリ秒単位）。たとえば、日時の文字列 Mon Feb 16 2015 00:00:00 GMT-0400 (Eastern Daylight Time) は、ミリ秒単位で 1424059200000 と表されます。
- toTimeInMilliSeconds：レポートの終了時間（ミリ秒単位）。たとえば、日時の文字列 Sun Feb 22 2015 00:00:00 GMT-0400 (Eastern Daylight Time) は、ミリ秒単位で 1424577600000 と表されます。

コード

```
public class GetResourceUsageCostSummaryTest {

 public static void main(String[] args)throws Exception {
  CuicServer api = CuicServer.getAPI("172.29.110.128",
"3E17CFFBA7A64C71B8958F40DE2EC9B3", "http", 80);
  UserAPIChargeBack instance = new UserAPIChargeBack(api);
  APIResourceUsageCostParams costParams = new APIResourceUsageCostParams();
  APINameValue value=new APINameValue();
  value.setName("vmid");
  value.setValue("56");
   List<APINameValue> requestParam=new ArrayList<APINameValue>();

   requestParam.add(value);
   costParams.setRequestParam(requestParam);
   costParams.setFromTimeInMilliSeconds(14429466000001);
   costParams.setToTimeInMilliSeconds(14430329990001);


  APIResourceUsageCostSummaryResponse response =
instance.userAPIGetResourceUsageCostSummary(costParams);
  System.out.println("Duration :" +response.getDuration());
  System.out.println("ResourceType :" +response.getResourceType());
  APINameValueList list=response.getResponseParamList();
  List<APINameValue> apivalue= list.getList();
  for(int i=0;i<apivalue.size();i++){
   System.out.println("Name :" + apivalue.get(i).getName());
   System.out.println("Value :" + apivalue.get(i).getValue());
  }

  List<APIResourceUsageCostSummary> summary=response.getResourceUsageCostSummary();
  for(int i=0;i<summary.size();i++){
   System.out.println("catalogItemName :" + summary.get(i).getCatalogItemName());
   //System.out.println("cpuResourceUsageCost" +
summary.get(i).getCpuResourceUsageCost());

   APICPUResourceUsageCost[] cpus=summary.get(i).getCpuResourceUsageCost();
   for(int j=0;j<cpus.length;j++){
   System.out.println("Cpu_GHz_Hours : " + cpus[j].getCpu_GHz_Hours());
   System.out.println("CPU CoreCost : " + cpus[j].getCpuCoreCost());
   System.out.println("Cpu Cores :" + cpus[j].getCpuCores());
   System.out.println("Cpu Cost :" + cpus[j].getCpuCost());
   System.out.println("Reserved CpuGHZCost" + cpus[j].getReservedCpuGhzCost());
   System.out.println("Used Cpu Cost" + cpus[j].getUsedCpuGhzCost());
   }
  }
 }
 }
```

結果

リクエストされたリソースの使用レポートが表示されます。

実装

userAPIGetResourceUsageCostSummary APIを使用して、名前と値の形式でレポートパラメータを渡し、リソースの使用レポートを表示します。

関連項目

月単位リソース使用レポートを確認するコードのサンプル：

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:{"requestParam":
[{"name":"vmid","value":"56"},{"name":"resourceName","value":"CPU"}],
"fromTimeInMilliSeconds":1441081800000,"toTimeInMilliSeconds":1443587400000}}
```

週単位リソース使用レポートを確認するコードのサンプル：

```
/app/api/rest?formatType=json&opName=
chargeback:userAPIGetResourceUsageCostSummary&opData={param0:{"requestParam":
[{"name":"vmid","value":"56"},{"name":"resourceName","value":"CPU"}],
"fromTimeInMilliSeconds":1442982600000,"toTimeInMilliSeconds":1443587400000}}
```

レポートの表示のための入力の組み合わせ候補：

- 月単位/週単位/日単位のタイムスタンプ、vmid、および resourceName（CPU、Disk、VM、BM、memory、または network）。
- 月単位/週単位/日単位のタイムスタンプ、vdcid、および resourceName（CPU、Disk、VM、BM、memory、または network）。
- 月単位/週単位/日単位のタイムスタンプ、Group、および resourceName（CPU、Disk、VM、BM、memory、または network）。
- 月単位/週単位/日単位のタイムスタンプ、vmid、resourceName なし。
- 月単位/週単位/日単位のタイムスタンプ、vdcid、resourceName なし。
- 月単位/週単位/日単位のタイムスタンプ、Group、resourceName なし。

使用可能なレポート定義の表示
履歴レポートの表示
スナップショット レポートの表示
表形式レポートの表示

スナップショット レポートの表示

目標

userAPIGetInstantDataReport API を使用して、レポート コンテキストのスナップショットを表示できます。レポート コンテキストでは、contextName、contextValue、および reportId が参照されます。

コンテキスト

レポート コンテキストのスナップショットを表示します。

前提条件

指定する contextName、contextValue、および reportId が Cisco UCS Director で使用可能であることが必要です。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIGetInstantDataReport&opData=
{param0:"1",param1:"Jha_Vmware_Cloud_82",param2:"VMS-ACTIVE-VS-INACTIVE-S0"}
```

応答

```
{ "serviceResult":{"categoryAxisName":null,"valueAxisName":"Active vs Inactive",
"categories":[{"categoryName":"","nameValuePairs":[{"name":"Active VMs","value":"42"},
{"name":"Inactive VMs","value":"29"}]}]}, "serviceError":null, "serviceName":"InfraMgr",

"opName":"userAPIGetInstantDataReport" }
```

コンポーネント

userAPIGetInstantDataReport API のパラメータは次のとおりです。

- String param0：レポート コンテキストの名前
- int param1：レポート コンテキストの値。
- int param2：レポートの固有識別子。

レポート コンテキストの名前と値の詳細については、『Cisco UCS Director Open Automation Cookbook』の付録 B：「Report Context Types and Report Context Names」を参照してください。

コード

```
import java.util.List;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.client.APISnapshotReport;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.client.ReportNameValuePair;
import com.cisco.cuic.api.client.SnapshotReportCategory;

public class TestuserAPIGetInstantDataReport {

 public static void main(String[] args) throws Exception {
  CuicServer server =
CuicServer.getAPI("172.29.110.222","96408900345D40C0BC889E4F41C2E094", "https", 443);

  UserAPIGlobal instance = new UserAPIGlobal(server);
  APISnapshotReport apiSanpshotReport=instance.userAPIGetInstantDataReport("1",
"Jha_Vmware_Cloud_82", "VMS-ACTIVE-VS-INACTIVE-S0");
  System.out.println("categoryAxisName: "+apiSanpshotReport.getCategoryAxisName());
  System.out.println("valueAxisName: "+apiSanpshotReport.getValueAxisName());
  List<SnapshotReportCategory> snapshotReportCategoryList=
apiSanpshotReport.getCategories();
  int i=0;
  for(SnapshotReportCategory snapshotReportCategory:snapshotReportCategoryList){
   System.out.println("Category_"+i);
   System.out.println("categoryName: "+snapshotReportCategory.getCategoryName());
   int j=0;
   ReportNameValuePair[]
reportNameValuePairArr=snapshotReportCategory.getNameValuePairs();
   for(ReportNameValuePair reportNameValuePair:reportNameValuePairArr ){
    System.out.println("ReportNameValuePair_"+j);
    System.out.println("name: "+reportNameValuePair.getName());
    System.out.println("value: "+reportNameValuePair.getValue());
    j++;
   }
   i++;
  }
 }
}
```

結果

指定したレポート コンテキストのスナップショットが表示されます。

実装

userAPIGetInstantDataReport API を使用して、contextName、contextValue、および reportId を渡し、指定したレポート コンテキストのスナップショットを表示します。

関連項目

使用可能なレポート定義の表示

履歴レポートの表示

リソース使用レポートの表示

表形式レポートの表示

表形式レポートの表示

目標

userAPIFilterTabularReport API を使用して、特定のコンテキストに基づいてフィルタリングされた表形式レポートのセットを表示できます。

コンテキスト

特定のコンテキストの表形式レポートを表示します。

前提条件

Cisco UCS Director でコンテキストが使用可能である必要があります。

REST の URL

要求

```
/app/api/rest?formatType=json&opName=userAPIFilterTabularReport&opData=
{param0:"1",param1:"VMware70",param2:"VMS-T0",param3:"VM-ID",param4:"1"}
```

応答

```
{ "serviceResult":{"rows":[{"Cloud":"VMware70","VM_ID":1,"User_Label":"",
"VM_Name":"vm-QA-SR169","Host_Name":null,"IP_Address":"","Image_Id":"vm-QA-SR169",
"Host_Node":"10.28.106.71","Power_Status":"ON","Group_Name":"Default Group",
"vDC":"test vdc","Category":"Discovered VM","Provisioned_Time":"",
"Scheduled_Termination_Time":"","Last_Status_Update":"May 18, 2016 02:35:32 UTC",
"Guest_OS_Type":"Red Hat Enterprise Linux 4 (32-bit)"}],"columnMetaData":null,
"reportParams":{}}, "serviceError":null, "serviceName":"InfraMgr",
"opName":"userAPIFilterTabularReport" }
```

コンポーネント

- param0：コンテキストの名前。
- param1：コンテキストの値。
- param2：レポートの ID。
- param3：列のラベル。
- param4：列の値。

コード

```
import com.cisco.cuic.api.client.APITabularReport;
import com.cisco.cuic.api.client.CuicServer;
import com.cisco.cuic.api.models.UserAPIGlobal;
import com.cisco.cuic.api.models.UserAPIServiceRequest;

public class TestuserAPIGetTabularReport
{
 public static void main(String[] args)throws Exception
{
 CuicServer api = CuicServer.getAPI("172.22.234.243",
"CF87FA987C8F4BBF814F2BB68CA6A823", "http", 80);
 UserAPIGlobal instance = new UserAPIGlobal(api);
 APITabularReport report=instance.userAPIFilterTabularReport("0", "All%20Clouds",
"VMS-T0","VM-ID","9");
 System.out.println(report.getRowCount());
 }
}
```

結果

特定のコンテキストの表形式レポートが、フィルタリングされて表示されます。

実装

userAPIFilterTabularReport API を使用して、レポートのパラメータを渡し、表形式レポートを表示します。

関連項目

使用可能なレポート定義の表示

履歴レポートの表示

リソース使用レポートの表示

スナップショット レポートの表示