vCloud SDK for PHP Developer's Guide

VMware vCloud SDK 1.0

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

EN-000363-00

vmware®

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

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

The VMware Web site also provides the latest product updates.

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

docfeedback@vmware.com

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

Contents

About This Book

This book, the *vCloud SDK for PHP Developer's Guide*, provides information about setting up your development environment to use the VMware vCloud® SDK for PHP.

VMware provides several APIs and SDKs for different applications and goals. This book provides information about using the vCloud SDK for PHP for developers who are creating PHP client applications for VMware vCloud Director.

To view the current version of this book as well as all VMware API and SDK documentation, go to http://www.vmware.com/support/pubs/sdk_pubs.html.

Revision History

This guide is revised with each release of the product or when necessary. A revised version can contain minor or major changes. Table 1 summarizes the significant changes in each version of this guide.

Table 1. Revision History

Revision	Description
01NOV10	Version 1.0. Amendments and minor corrections to Beta content.
31AUG10	Version 1.0 (Beta)

Intended Audience

This guide is intended for software developers who are building vCloud API applications, including interactive clients of VMware Cloud Director. This guide assumes you are familiar with the PHP programming language, Representational State Transfer (REST) and RESTful programming conventions, the Open Virtualization Format Specification (OVFS), and VMware virtual machine technology. Familiarity with other technologies such as XML, HTTP, and the Windows or Linux operating systems is also assumed.

VMware Technical Publications Glossary

VMware Technical Publications provides a glossary of terms that might be unfamiliar to you. For definitions of terms as they are used in VMware technical documentation go to http://www.vmware.com/support/pubs.

Document Feedback

VMware welcomes your suggestions for improving our documentation. Send your feedback to docfeedback@vmware.com.

Technical Support and Education Resources

The following sections describe the technical support resources available to you. To access the current versions of other VMware books, go to http://www.vmware.com/support/pubs.

Online and Telephone Support

To use online support to submit technical support requests, view your product and contract information, and register your products, go to http://communities.vmware.com/community/developer.

Support Offerings

To find out how VMware support offerings can help meet your business needs, go to http://www.vmware.com/support/services.

VMware Professional Services

VMware Education Services courses offer extensive hands-on labs, case study examples, and course materials designed to be used as on-the-job reference tools. Courses are available onsite, in the classroom, and live online. For onsite pilot programs and implementation best practices, VMware Consulting Services provides offerings to help you assess, plan, build, and manage your virtual environment. To access information about education classes, certification programs, and consulting services, go to http://www.vmware.com/services.

About the vCloud SDK for PHP 1

The VMware vCloud API provides support for developers who are building interactive clients of VMware vCloud Director using a RESTful application development style. vCloud API clients and servers communicate over HTTP, exchanging representations of vCloud objects. These representations take the form of XML elements. HTTP GET requests are used to retrieve the current representation of an object, HTTP POST and PUT requests are used to create or modify an object, and HTTP DELETE requests are typically used to delete an object.

The vCloud SDK for PHP is a PHP language binding for the vCloud API. It provides a PHP class library and a set of example applications. The classes and functions in the library encapsulate the interfaces, objects, and operations that the vCloud API supports, while preserving its RESTful approach and compatibility with the HTTP protocol family.

This *vCloud SDK for PHP Developer's Guide* provides information about setting up the SDK in a development environment, and information about running the example applications included in the SDK.

This chapter includes the following topics:

vCloud Object Taxonomy

The vCloud SDK for PHP defines a set of objects common to cloud computing environments. Figure 1-1 shows the principal object types.

vCloud Organizations

A vCloud contains one or more organizations. A vCloud organization is a unit of administration for a collection of users, groups, and computing resources. Users authenticate at the organization level, supplying credentials established by an organization administrator when the user was created or imported.

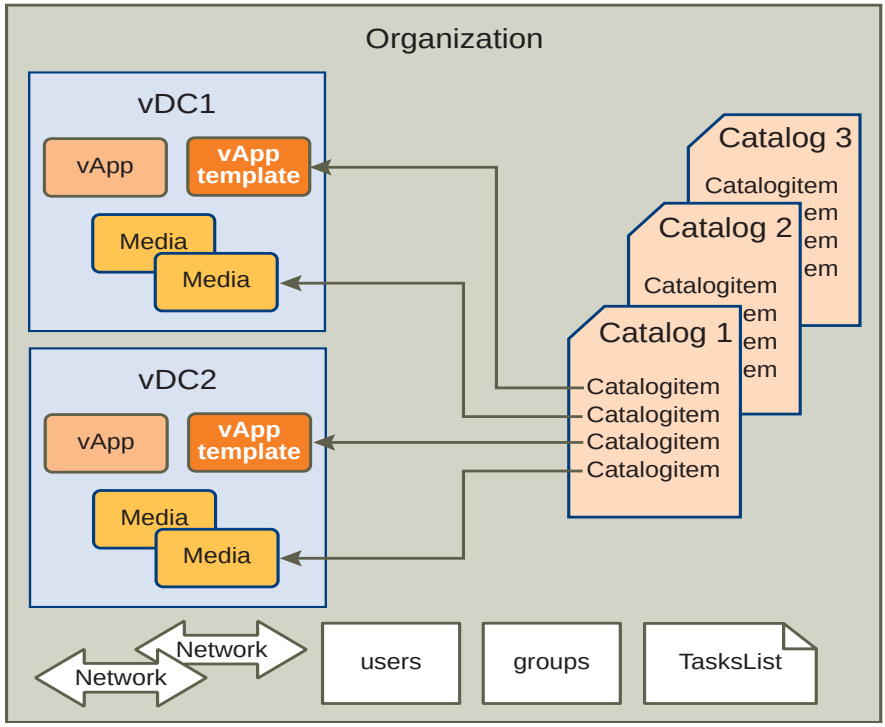
vCloud Users and Groups

An organization can contain an arbitrary number of users and groups. Users can be created by the organization administrator or imported from a directory service such as LDAP. Groups must be imported from the directory service. Permissions within an organization are controlled through the assignment of rights and roles to users and groups.

vCloud Networks

An organization can be provisioned with one or more networks. These organization networks can be configured to provide services such as DHCP, NAT, and firewalls.

Figure 1-1. vCloud Object Taxonomy



vCloud Virtual Datacenters

A vCloud virtual datacenter (vDC) is an allocation mechanism for resources such as networks, storage, CPU, and memory. In a vDC, computing resources are fully virtualized, and can be allocated based on demand, service level requirements, or a combination of the two.

There are two kinds of vDCs:

- **Provider vDCs**. These vDCs contain all the resources available from the vCloud service provider. Provider vDCs are created and managed by vCloud system administrators.
- **Organization vDCs**. These vDCs provide an environment where virtual systems can be stored, deployed, and operated. They also provide storage for virtual media, such as floppy disks and CD ROMs.

An organization administrator specifies how resources from a provider vDC are distributed to the vDCs in an organization.

vCloud Catalogs

Catalogs contain references to virtual systems and media images. A catalog can be shared to make it visible to other members of an organization, and can be published to make it visible to other organizations. A vCloud system administrator specifies which organizations can publish catalogs, and an organization administrator controls access to catalogs by organization members.

vCloud Tasks

Long-running operations initiated by members of an organization create tasks, which are kept on the organization's tasks list.

Virtual Systems and Media Images in a vCloud

Virtual systems and media images are stored in a vDC and can be included in a catalog. Media images are stored in their native representation (ISO or floppy). Virtual systems are stored as templates, using an open standard format (OVF 1.0). These templates can be retrieved from catalogs and transformed into virtual systems, called vApps, through a process called instantiation, which binds a template's abstract resource requirements to resources available in a vDC. A vApp contains one or more individual virtual machines (`Vm` elements), along with parameters that define operational details such as:

- How the contained virtual machines are connected to each other and to external networks.
- The order in which individual virtual machines are powered on or off.
- End-user license agreement terms for each virtual machine.
- Deployment lease terms (typically inherited from the containing organization) that constrain the vApp's consumption of vDC resources.
- Access control information specifying which users and groups can perform operations such as deploy, power on, modify, and suspend on the vApp and the virtual machines it contains.

vCloud SDK for PHP Design

The vCloud SDK for PHP includes the following packages:

- API packages, listed in Table 1-1. These packages contain classes that represent complex types defined in vCloud API, vCloud administrative API and vCloud vSphere platform API extensions. Classes in this package are generated from the vCloud API XML schema files. Each class maps to a complex type defined in those files. Objects of these classes are referred to as vCloud data objects.

  Table 1-1. VMware_VCloud_API Packages

Package Name	Package Contents
VMware_VCloud_API	Classes representing objects defined in the vCloud user API and administrative API
VMware_VCloud_API_OVF	Classes representing objects defined in the OVF specification
VMware_VCloud_API_Extension	Classes representing objects defined in the vCloud API vSphere Platform Extensions
VMware_VCloud_API_Version	Classes representing objects that contain vCloud API version information

- SDK packages, listed in Table 1-2. These packages contain classes that implement vCloud API operations. Each of the classes maps to a vCloud resource entity. Classes manage the resource entity lifecycle (create, retrieve, update, and delete, often abbreviated as CRUD). This package also implements various utility functions associated with connecting to a vCloud instance, marshalling requests, unmarshalling responses, and so on. Objects of these classes are referred to as vCloud SDK objects.

  Table 1-2. VMware_VCloud_SDK Packages

Package Name	Package Contents
VMware_VCloud_SDK	Classes that implement operations defined in the vCloud user API and administrative API
VMware_VCloud_SDK_Extension	Classes that implement operations defined in the vCloud API vSphere Platform Extensions
VMware_VCloud_SDK_HTTP	Classes that support HTTP client operations.

Creating a Data Object

Each data object class includes a constructor method whose parameters represent attributes of the class and all of its ancestors. Attributes are marked as protected to restrict their visibility. All classes contain setter and getter methods for XML elements and attributes. The general form of these method names is *operation_attribute-name* for attributes and *operationElementName* for elements, where *operation* is one of `set` or `get`. For example, the `VMware_VCloud_API_ReferenceType` class supports `set_name()` and `get_name()` methods that get or set the value of its `name` attribute, and the `VMware_VCloud_API_UserType` class supports `setFullName()` and `getFullName()` methods that set or get the value of the `FullName` element in a `User` object.

To create a data object, you can either invoke an empty constructor and then call the setters for the object (see Example 1-1), or invoke the constructor with parameters (see Example 1-2).

Example 1-1. Create a Data Object From an Empty Constructor

```
$ref = new VMware_VCloud_API_ReferenceType();
    $ref->set_href($href);
    $ref->set_type($type);
    $ref->set_name($name);
```

Example 1-2. Create a Data Object With a Parameterized Constructor

```
$ref = new VMware_VCloud_API_ReferenceType ($href=$href, $type=$type, $name=$name);
```

Creating an SDK Object

You can create an SDK object when you need to invoke a lifecycle operation such as create or modify on a vCloud API object. Most class constructors for SDK objects require two parameters:

- A `VMware_VCloud_SDK_Service` object, which contains HTTP connection information.
- A `ReferenceType` data object which contains the request URL. For more information about request URLs, see the *vCloud API Programming Guide*.

For example, you could use code similar to the fragment shown in Example 1-3 to create a VMware_VCloud_SDK_Org object to use as an entrypoint for client operations.

Example 1-3. Creating an SDK Object

```
// get the list of all organizations in the vCloud
$orgRefs = $service->getOrgRefs($orgName);
// create an object that represents the first organization in the list
$sdkOrg = $service->createSDKObj($orgRefs[0]);
```

Table 1-3 summarizes the types of SDK objects you can create and provides information to help you choose the creation parameters (the container for the object, and a method to use). The table omits the VMware_VCloud_SDK_ part of the package names in the **SDK Object** and **Container Object** columns.

To use the information in the table to create an SDK object

1 Retrieve an array of object references by specifying a container object and a creation method.

   $*references*=*Column2*->*Column3*

2 For any reference in the array, create an SDK object using the selected reference.

   Column1=$service->createSDKObj($*reference*)

Example 1-3 applies this formula to creating an `Org` object.

NOTE Rows in Table 1-3 where SDK Object is listed as None indicate operations that return a read-only object, such as a `RightReference`, that may be needed when creating other objects.

Table 1-3. Summary of SDK Objects, Containers, and Methods

SDK Object	Container Object	Object Reference Creation Method
None	Admin	getRightRefs()
None	Admin	getProviderVdcRefs()
None	Extension_VMWProviderVdc	getNetworkPoolRefs()
None	Extension_VimServer	getResourcePoolRefs()
Admin	None	See "Creating Top-Level Objects" on page 12
AdminCatalog	AdminOrg	getAdminCatalogRefs()
AdminNetwork	AdminOrg	getAdminNetworkRefs()
AdminOrg	Admin	getAdminOrgRefs()
AdminVdc	AdminOrg	getAdminVdcsRefs()
Catalog	Org	getCatalogRefs()
CatalogItem	Catalog	getCatalogItemRefs()
CatalogItem	AdminCatalog	getCatalogItemRefs()
Extension	None	See "Creating Top-Level Objects" on page 12
Extension_Host	Extension	getHostRefs()
Extension_VimServer	Extension	getVimServerRefs()
Extension_VMWExternalNetwork	Extension	getVMWExternalNetworkRefs()
Extension_VMWNetworkPool	Extension	getVMWNetworkPoolRefs()
Extension_VMWProviderVdc	Extension	getVMWProviderVdcRefs()
ExternalNetwork	Admin	getExternalNetworkRefs()
Group	AdminOrg	getGroupRefs()
Media	Vdc	getMediaRefs()
Network	Org	getOrgNetworkRefs()
Org	Service	getOrgRefs(), getSystemOrgRef(), getAdminSystemOrgRef()
Role	Admin	getRoleRefs()
Service	None	See "Creating Top-Level Objects" on page 12
User	AdminOrg	getUserRefs()
VApp	Vdc	getVAppRefs()
VApp	VApp	getContainedVAppRefs()
VAppTemplate	Vdc	getVAppTemplateRefs()
Vdc	Org	getVdcRefs()
Vm	VApp	getContainedVmRefs()

Creating Top-Level Objects

The vCloud SDK for PHP provides methods that you can use to create a reference to a top level vCloud API object such as `VCloud` and `VMWExtension`. These objects do not have containers, so you cannot use the more common method shown in Example 1-3. Instead, use one of the following:

- To create a `VMware_VCloud_SDK_Service` object to use as an entrypoint for user API operations:

  ```
  $service = VMware_VCloud_SDK_Service::getService();
  ```

- To create a `VMware_VCloud_SDK_Admin` object to use as an entrypoint for administrative operations:

  ```
  $sdkAdminObj = $service->createSDKAdminObj();
  ```

- To create a `VMware_VCloud_SDK_Extension` object to use as an entrypoint for vSphere Platform Extensions operations:

  ```
  $sdkExtObj = $service->createSDKExtensionObj();
  ```

Using a Different HTTP Library

Example programs included in the vCloud SDK for PHP require the `PEAR HTTP_Request2` package. To use a different HTTP library, create an HTTP client object that implements the `VMware_VCloud_SDK_Http_Client_Interface` interface, and then can call the `VMware_VCloud_SDK_Service::getService()` method specifying that client.

```
$service = VMware_VCloud_SDK_Service::getService($myHTTPClient);
```

Using the HTML Reference Material

The reference documentation in the in the `docs` folder of the vCloud SDK for PHP download provides detailed information on classes and functions.

To use the HTML documentation

1. Open the `docs` folder in the download and open the file `index.html` in a browser.
2. Select **VMware_VCloud_API** from the **Packages** drop-down menu.
3. Select a class (for example, `VMware_VCloud_API_AdminOrgType`) in the left-hand pane.
4. Go to the **Method Summary** section in the right-hand pane and click the link for the `__construct()` method.

   The method summary lists the constructors for required and optional attributes and elements of the class, sorted by type. You can click the name of any element, then go to its method summary to get information about its constructors. For example, `VMware_VCloud_API_AdminOrgType` requires a `VMware_VCloud_API_OrgSettingsType` element. You can click the element name to see its method summary, and click its `__construct()` method to see the details of how to construct it.

Setting Up for PHP Development 2

This chapter describes how to prepare for using the vCloud SDK for PHP, and how to download the SDK and use it.

This chapter includes the following topics:

Prerequisites

The vCloud SDK for PHP requires the following software to be installed on the computer where you install the SDK:

- PHP 5.3.1 or later.
- The PEAR HTTP_Request2 package, version 0.5.1, available from (http://pear.php.net/package/HTTP_Request2/download.

This document and the SDK reference documentation assume that you are familiar with the PHP programming language and have access to an installation of VMware vCloud Director. In addition, you should consider the following:

- Although the vCloud SDK for PHP reference documentation provides information about the vCloud API XML schemas, which define the objects and operations that the SDK supports, familiarity with the details of the underlying objects and operations, as described in the *vCloud API Programming Guide*, can help you understand the structure of vCloud API objects, and how the methods in this SDK operate on those objects.
- Before you can run the examples, you must use the vCloud Director web console or the vCloud API to create an organization, catalog, and vDC that the examples can use. The organization must have a user account with rights to run the examples. The predefined `CatalogAuthor` role should provide all the necessary rights. For more information about roles and rights, see the *VMware Cloud Director Administrator's Guide.*
- Several of the example programs require you to have an OVF package available on the client host. This package must be uncompressed, and can specify one or more `vmdk` files. For more information about OVF, see the *vCloud API Programming Guide.*

Download and Install the vCloud SDK for PHP

You can download the vCloud SDK for PHP from the VMware website. The SDK is distributed as a compressed archive in two forms.

- `vcloudPHP_1.0.0.`*build*`.tar.gz`, a compressed archive in `tar` format, where *build* is a build number.
- `vcloudPHP_1.0.0.`*build*`.zip`, a compressed archive in `zip` format.

To download and install the vCloud SDK for PHP

1. In a browser, go to http://www.vmware.com/go/vcloudsdkforphp.
2. In the **Resources** area of the vCloud SDK for PHP Community page, click the **Download** button.
3. On the **Download** page, log in with your VMware customer credentials.
4. Review the license agreement. Click **Yes** to accept it and continue with the download, or click **No** to exit without downloading.
5. Choose a download option, then click the link for the distribution format you want. The vCloud SDK for PHP is distributed as a compressed archive in two forms.
   - `vcloudPHP_1.0.0.`*build*`.tar.gz`, a compressed archive in `tar` format, where *build* is a build number.
   - `vcloudPHP_1.0.0.`*build*`.zip`, a compressed archive in `zip` format.
6. When the download completes, uncompress the download package into any convenient folder on your computer. Uncompressed, either archive requires about 32 MB of disk space. The package includes the following folders:
   - `docs`: vCloud SDK for PHP reference documentation (*vCloud SDK for PHP Reference Guide*) in HTML format.
   - `library`: A collection of class libraries and functions that encapsulate vCloud API objects and operations.
   - `samples`: Example code demonstrating common use cases associated with programmatically managing virtual infrastructure.

Hello vCloud: A Structured Workflow Example

3

This chapter presents an example of using the vCloud SDK for PHP to implement a structured workflow through the lifecycle of a vApp.

The `helloVCloud.php` example, included in the `samples` folder of the SDK, demonstrates these operations supported by the vCloud SDK for PHP:

- Logging in to a vCloud organization
- Browsing the organization to find a vDC and a catalog
- Instantiating a vApp template from the catalog to create a vApp
- Operating the vApp
- Logging out

Like all example the programs, `helloVCloud.php` is liberally commented. Read the comments for more information about how this example uses the features of the vCloud SDK for PHP. Unlike the other example programs, `helloVCloud.php` does not read its runtime options from the file `config.php`. You must supply runtime options on the command line. To see a summary of `helloVCloud.php` options, use the following command:

```
php helloVCloud.php --help
```

To run the `helloVCloud.php` example, use the following command:

```
php helloVCloud.php -s server -u user@vcloudOrganization -p password -c config -o=orgName
-d=vdcName -g=catalogName -i=item -a=vAppName
```

The following options are required:

- *server* is the hostname or IP address of a vCloud Director server.
- *username* is the name of a vCloud Director user, in the form *user@vcloudOrganization*, who has rights to create and operate vApps.
- *password* is the user's password.

The following options are optional:

- *config* is a set of HTTP connection parameters in the form of a PHP array. If you omit this option, `helloVCloud.php` accepts any server certificate. The following specification of config enables certificate validation, using a certificate stored in `/tmp/cert.crt`:

  ```
  -c='ssl_verify_peer=>true, ssl_verify_host=>true, ssl_cafile=>/tmp/cert.crt'
  ```

- *orgName* is the name of the organization to which the user is authenticating.
- *vdcName* is the name of a vDC in that organization where the user can instantiate and deploy the vApp.
- *catalogName* is the name of a catalog in the user's organization.

- *item* is value of the `name` attribute of a CatalogItem element that references the vApp template you plan to instantiate. This CatalogItem must be contained by the catalog specified by *catalogName*.
- *vAppName* is the name to give to the vApp that this sample creates.

All options but `–s`, `–u`, and `–p` must be separated from their arguments by an equals sign. For example:

```
php helloVCloud.php –s vcloud.example.com –u user@exampleOrg –p Pa55w0rd –o=exampleOrg
–d=exampleVdc –g=exampleCatalog –i=exampleTemplate –a=MyVapp
```

You can use the vCloud Director Web Console or the vCloud REST API to find appropriate values in your vCloud for *orgName, vdcName, catalogName*, and *item*. See the *Cloud Director User's Guide*, or the chapter on browsing in the *vCloud API Programming Guide.*

Using the Example Programs 4

In addition to `helloVCloud.php`, the vCloud SDK for PHP includes a number of other example programs that demonstrate how to use the SDK to develop client applications. The examples are in the `samples` folder of the SDK download.

Comments in the examples provide detailed information about how they use the features of the vCloud SDK for PHP.

This chapter includes the following topics:

Configuring a Runtime Environment for the Examples

The `samples` folder includes a file named `config.php` that provides values for parameters used in the examples. These parameters include the credentials that the examples use for logging in, names for objects such as catalogs and vDCs that the examples create, and other values that you are required to specify when creating a vCloud API object. You must edit this file before you can run any of the examples. Some parameters are initialized to a default value. Parameters that you are required to set are initialized with a place-holder value of `please set`. Comments in the file provide more information about the parameters and how to obtain appropriate values for them.

Running the Examples

Some of the example programs require system administrator privileges to run. Others can be run by any user who can create and operate a vApp.

To run any of the examples

1 Edit `config.php` to provide required parameter values.

2 Run the example in a shell window using a command of the form, where *example* is the name of the example program:

```
php example.php
```

Table 4-1 lists the example programs that do not require system administrator privileges to run, and provides brief descriptions of what they do.

Table 4-1. Example Programs That Do Not Require System Administrator Privileges

Example Name	Description
addCatalogItem.php	Adds an item to a catalog
config.php	Provides parameter values for all examples
createCatalog.php	Creates a catalog
deployPowerOnVApp.php	Deploys and powers on a vApp
helloVCloud.php	A structured workflow example that uses command-line parameters
instantiateVAppTemplateDefault.php	Instantiates a vApp template using organization defaults
login.php	Authenticates a user
sampleHelper.php	Used by all example code
updateVmMemory.php	Edits the memory required by a virtual machine and reduces the existing value by half
uploadVAppTemplate.php	Uploads an OVF package to create a vApp template.

Table 4-2 lists the example programs that require system administrator privileges to run.

Table 4-2. Example Programs That Require System Administrator Privileges

Example Name	Description
addOrgNetwork.php	Adds a network to an organization
createAdminOrg.php	Creates an organization
createAdminVdc.php	Creates a vDC
extCreateExternalNetwork.php	Creates an external network from vSphere resources
extCreateNetworkPool_PG.php	Creates a network pool from vSphere resources
extCreateProviderVdc.php	Creates a provider vDC from vSphere resources
extDisableVimServer.php	Disables a vCenter server registered for use with vCloud Director
extImportVmAsVApp.php	Imports a virtual machine from vCenter to create a vApp in the specified vDC
extPrepareHost.php	Prepares an ESX/ESXi host for use with vCloud Director
extRegisterVimServer.php	Registers a vCenter server for use with vCloud Director

Index