Codierungsrichtlinien für WFA

OnCommand Workflow Automation

NetApp
October 09, 2025

Inhalt

Codierungsrichtlinien für WFA

Sie sollten die allgemeinen Richtlinien zur Kodierung von OnCommand Workflow Automation (WFA), Namenskonventionen und Empfehlungen zum Erstellen verschiedener Bausteine wie Filter, Funktionen, Befehle und Workflows verstehen.

Richtlinien für Variablen

Bei der Erstellung eines Befehls oder eines Datenquelltyps müssen Sie die Richtlinien für PowerShell und Perl-Variablen in OnCommand Workflow Automation (WFA) kennen.

PowerShell Variablen

Richtlinien	Beispiel

Für Skript-Eingabeparameter:
• Verwenden Sie Pascal Case.
• Keine Unterstriche verwenden.
• Verwenden Sie keine Abkürzungen.
	`$VolumeName`
	`$AutoDeleteOptions`
	`$Size`

Für interne Skriptvariablen:
• Verwenden Sie die Camel-Hülle.
• Keine Unterstriche verwenden.
• Verwenden Sie keine Abkürzungen.
	`$newVolume`
	`$qtreeName`
	`$time`

Für Funktionen: • Verwenden Sie Pascal Case. • Keine Unterstriche verwenden. • Verwenden Sie keine Abkürzungen.	`GetVolumeSize`
Bei Variablennamen wird die Groß-/Kleinschreibung nicht beachtet. Um die Lesbarkeit zu verbessern, sollten Sie jedoch für denselben Namen keine andere Großschreibung verwenden.	`$variable` Ist das gleiche wie `$Variable.`
Variablennamen sollten in einfachem Englisch sein und sich auf die Funktionalität des Skripts beziehen.	Nutzung `$name` Und nicht `$a.`
Erklären Sie den Datentyp für jede Variable explizit.	[String]Name [Int]Größe

Richtlinien	Beispiel
Verwenden Sie keine Sonderzeichen (! @ # & % , .) und Leerzeichen.	Keine
Verwenden Sie keine PowerShell reservierten Schlüsselwörter.	Keine
Gruppieren Sie die Eingabeparameter, indem Sie zunächst die obligatorischen Parameter, gefolgt von den optionalen Parametern platzieren.	`param( [parameter(Mandatory=$true)] [string]$Type,  [parameter(Mandatory=$true)] [string]$Ip,  [parameter(Mandatory=$false)] [string]$VolumeName ) `
Kommentieren Sie alle Eingabevariablen mit *`HelpMessage`* Anmerkung mit einer aussagekräftigen Hilfemeldung.	`[parameter(Mandatory=$false,HelpMessage="LUN to map")] [string]$LUNName`
Verwenden Sie „`Filer`“ nicht als Variablenname, sondern verwenden Sie „`Array`“.	Keine
Nutzung *`ValidateSet`* Anmerkung in Fällen, in denen das Argument aufzählen wird. Damit wird automatisch der Datentyp „Enum“ für den Parameter angezeigt.	`[parameter(Mandatory=$false,HelpMessage="Volume state")] [ValidateSet("online","offline","restricted")] [string]$State`

Richtlinien	Beispiel

Fügen Sie einem Parameter, der mit „`_Capacity`“ endet, einen Alias hinzu, um anzugeben, dass der Parameter vom Kapazitätstyp ist.	Der Befehl „`Create Volume`“ verwendet Aliase wie folgt:
	`[parameter(Mandatory=$false,HelpMessage="Volume increment size in MB")]`
	`[Alias("AutosizeIncrementSize_Capacity")]`
	`[int]$AutosizeIncrementSize`

Fügen Sie einem Parameter, der mit „`_Password`“ endet, einen Alias hinzu, um anzugeben, dass der Parameter einen Kennworttyp hat.	`param (`
	`  [parameter(Mandatory=$false, HelpMessage="In order to create an Active Directory machine account for the CIFS server or setup CIFS service for Storage Virtual Machine, you must supply the password of a Windows account with sufficient privileges")]`
	`[Alias("Pwd_Password")]`
	`[string]$ADAdminPassword`
	`)`

Perl-Variablen

Richtlinien	Beispiel

Für Skript-Eingabeparameter:
• Verwenden Sie Pascal Case.
• Keine Unterstriche verwenden.
• Verwenden Sie keine Abkürzungen.
	`$VolumeName`
	`$AutoDeleteOptions`
	`$Size`

Verwenden Sie keine Abkürzungen für interne Skriptvariablen.	`$new_volume`
	`$qtree_name`
	`$time`

Verwenden Sie keine Abkürzungen für Funktionen.	`get_volume_size`

Richtlinien	Beispiel
Bei Variablennamen wird die Groß-/Kleinschreibung beachtet. Um die Lesbarkeit zu verbessern, sollten Sie für denselben Namen keine andere Groß-/Kleinschreibung verwenden.	`$variable` Ist nicht dasselbe wie `$Variable.`
Variablennamen sollten in einfachem Englisch sein und sich auf die Funktionalität des Skripts beziehen.	Nutzung `$name` Und nicht `$a.`
Gruppieren Sie die Eingabeparameter, indem Sie zuerst die obligatorischen Parameter, gefolgt von den optionalen Parametern platzieren.	Keine
In GetOptions Funktion, deklarieren Sie explizit den Datentyp jeder Variable für Eingabeparameter.	`GetOptions( "Name=s"=>\$Name, "Size=i"=>\$Size )`
Verwenden Sie „`Filer`“ nicht als Variablenname, sondern verwenden Sie „`Array`“.	Keine
Perl schließt nicht das ein *`ValidateSet`* Anmerkung für Aufzählungswerte. Verwenden Sie die expliziten „`if`“-Anweisungen für Fälle, in denen das Argument aufgezählte Werte erhält.	`if (defined$SpaceGuarantee&&!($SpaceGuaranteeeq'none'`
	$SpaceGuaranteeeq’volume'
	$SpaceGuaranteeeq’file')) { die’Illegal SpaceGuarantee argument: \".$SpaceGuarantee.'\"; } ----
Alle Perl WFA Befehle müssen das Pragma „`strict`“ verwenden, um die Verwendung unsicherer Konstrukte für Variablen, Referenzen und Unterroutinen zu entmutigen.	`use strict; # the above is equivalent to use strictvars; use strictsubs; use strictrefs;`

Richtlinien	Beispiel

Alle Perl WFA Befehle müssen die folgenden Perl Module verwenden:
• Getopt
Dies wird zur Angabe von Eingabeparametern verwendet.
• WFAUtil
Dies wird für Dienstprogrammfunktionen verwendet, die für die Protokollierung von Befehlen, für die Meldung des Befehlsfortschritts, für die Verbindung zu Array-Controllern usw. bereitgestellt werden.
	`use Getopt::Long;`
	`use NaServer;`
	`use WFAUtil;`

Richtlinien für Einzüge

Beim Schreiben eines PowerShell oder Perl Skripts für OnCommand Workflow Automation (WFA) müssen Sie die Richtlinien zum Einbinden kennen.

Richtlinien	Beispiel
Eine Registerkarte entspricht vier leeren Leerzeichen.	

Verwenden Sie Laschen und Klammern, um den Anfang und das Ende eines Blocks anzuzeigen.	PowerShell Skript
	`if`
	`($pair.length-ne 2)`
	`{`
	`throw "Got wrong input data"`
	`}`
	Perl-Skript
	`if`
	`(defined $MaxDirectorySize)`
	`{`
	`# convert from MBytes to Bytes`
	`my $MaxDirectorySizeBytes =`
	`$MaxDirectorySize *`
	`1024 * 1024;`
	`}`

Richtlinien	Beispiel
Fügen Sie leere Zeilen zwischen Gruppen von Vorgängen oder Codebrocken hinzu.	``` $options=$option.trim(); $pair=$option.split(" "); Get-WFAlogger -Info -messages $("split options: "+ $Pair) ```

Richtlinien für Kommentare

In Ihren Skripten für OnCommand Workflow Automation (WFA) müssen Sie die Richtlinien für PowerShell und Perl Kommentare kennen.

PowerShell kommentiert

Richtlinien	Beispiel
Verwenden Sie das Zeichen # für einen einzelnen Zeilenkommentar.	``` # Single line comment $options=$option.trim(); ```
Verwenden Sie das Zeichen # für einen Zeilenendkommentar.	``` $options=$option.trim(); # End of line comment ```
Verwenden Sie die Zeichen <# und #> für einen Blockkommentar.	``` <# This is a block comment #> $options=$option.trim(); ```

Perl-Kommentare

Richtlinien	Beispiel

Verwenden Sie das Zeichen # für einen einzelnen Zeilenkommentar.	`# convert from MBytes to Bytes`
	`my $MaxDirectorySizeBytes =`
	`$MaxDirectorySize *`
	`1024 * 1024;`

Verwenden Sie das Zeichen # für den Zeilenendkommentar.	`my $MaxDirectorySizeBytes =`
	`$MaxDirect`
	`orySiZe * 1024 * 1024; # convert`
	`to Bytes`

Verwenden Sie das # Zeichen in jeder Zeile mit einem leeren # am Anfang und am Ende, um einen Kommentarrahmen für mehrzeilige Kommentare zu erstellen.	`#`
	`# This is a multi-line comment.`
	`Perl 5, unlike`
	`# Powershell, does not have direct`
	`support for`
	`# multi-line comments. Please use`
	`a '#'in every line`
	`# with an empty '#' at the`
	`beginning and end to create`
	`# a comment border`
	`#`

Fügen Sie in den WFA Befehlen keinen kommentierten und toten Code ein. Zu Testzwecken können Sie jedoch den Mechanismus der Plain Old Documentation (POD) verwenden, um den Code zu kommentieren.	`=begin comment`
	`# Set deduplication`
	`if(defined $Deduplication &&`
	`$Deduplication eq "enabled")`
	`{`
	`$wfaUtil-`
	`>sendLog("Enabling`
	`Deduplication");`
	`}`
	`=end comment`
	`=cut`

Richtlinien für die Protokollierung

Sie müssen die Richtlinien für die Protokollierung beim Schreiben eines PowerShell oder

Perl Skripts für OnCommand Workflow Automation (WFA) kennen.

PowerShell-Protokollierung

Richtlinien	Beispiel
Verwenden Sie das Cmdlet "Get-WFALogger" zur Protokollierung.	`Get-WFALogger -Info -message “Creating volume”`
Protokollieren jeder Aktion, die Interaktion mit internen Paketen wie Data ONTAP, VMware und PowerCLI erfordert.Alle Protokollmeldungen stehen in Ausführungsprotokollen im Ausführungsstatus-Verlauf von Workflows zur Verfügung.	Keine
Protokollieren Sie alle relevanten Argumente, die an interne Pakete übergeben werden.	Keine
Verwenden Sie je nach Nutzungskontext die entsprechenden Protokollebenen, wenn Sie das Cmdlet "Get-WFALogger" verwenden. -Info, -Error, -warn und -Debug sind die verschiedenen verfügbaren Protokollebenen. Wenn keine Protokollebene angegeben wird, ist die Standard-Protokollebene Debug.	Keine

Perl-Protokollierung

Richtlinien	Beispiel
Verwenden Sie das WFAUtil sendLog zur Protokollierung.	`my wfa_util = WFAUtil->new(); eval { $wfa_util->sendLog('INFO', "Connecting to the cluster: $DestinationCluster"); }`
Protokollieren jeder Aktion, die eine Interaktion mit anderen externen Aktionen wie Data ONTAP, VMware und WFA erfordert Alle Log-Nachrichten, die Sie mit der WFAUtil sendLog-Routine erstellen, werden in der WFA-Datenbank gespeichert. Diese Protokollmeldungen stehen für den ausgeführten Workflow und Befehl zur Verfügung.	Keine

Richtlinien	Beispiel
Protokollieren Sie alle relevanten Argumente, die an die Routine übergeben wurden, die aufgerufen wurde.	Keine
Verwenden Sie die entsprechenden Protokollebenen.-Info, -Error, -warn und -Debug sind die verschiedenen verfügbaren Protokollebenen.	Keine

Wenn Sie sich auf der -Info-Ebene anmelden, seien Sie präzise und präzise. Geben Sie keine Implementierungsdetails wie Klassenname und Funktionsname in Protokollmeldungen an. Beschreiben Sie den genauen Schritt oder den genauen Fehler in einfachem Englisch.	Der folgende Code-Snippet zeigt ein Beispiel für eine gute Nachricht und eine schlechte Nachricht:
	`$wfa_util->sendLog('WARN', "Removing volume: '.$VolumeName); # Good Message`
	`$wfa_util->sendLog('WARN', 'Invoking volume-destroy ZAPI: '.$VolumeName); # Bad message`

Richtlinien für die Fehlerbehandlung

Beim Schreiben eines PowerShell oder Perl Skripts für OnCommand Workflow Automation (WFA) müssen Sie die Richtlinien für die Fehlerbehandlung kennen.

PowerShell Fehlerbehandlung

Richtlinien	Beispiel

Zu den Cmdlets durch PowerShell Runtime wurden allgemeine Parameter wie ErrorAction und WarningAction hinzugefügt.
• Der Parameter ErrorAction bestimmt, wie ein Cmdlet auf einen nicht-terminenden Fehler des Befehls reagieren soll.
• Der Parameter WarningAction bestimmt, wie ein Cmdlet auf eine Warnung aus dem Befehl reagieren soll.
• Stopp, SilentlyWeiter, Abfragen und Fortfahren sind die gültigen Werte für die Parameter ErrorAction und WarningAction.
Weitere Informationen finden Sie unter `Get-Help about_CommonParameters` Befehl in PowerShell CLI.
	ErrorAction: Das folgende Beispiel zeigt, wie ein nicht-terminierender Fehler als Fehler beim Beenden behandelt wird:
	`New-NcIgroup-Name $IgroupName-Protocol $Protocol-Type$OSType-ErrorActionstop`
	WarningAction
	`New-VM-Name $VMName-VM $SourceVM-DataStore$DataStoreName-VMHost$VMHost-WarningActionSilentlyContinue`

Verwenden Sie die allgemeine Anweisung „`Try/Catch`“, wenn der Typ der eingehenden Ausnahme nicht bekannt ist.	`try`
	`{`
	`"In Try/catch block"`
	`}`
	`catch`
	`{`
	`"Got exception"`
	`}`

Verwenden Sie die spezifische „`Try/Catch`“-Anweisung, wenn der Typ der eingehenden Ausnahme bekannt ist.	`try`
	`{`
	`"In Try/catch block"`
	`}`
	`catch[System.Net.WebExceptional],`
	`[System.IO.`
	`IOException]`
	`{`
	`"Got exception"`
	`}`

Richtlinien	Beispiel

Verwenden Sie die Anweisung „`endlich`“, um Ressourcen freizugeben.	```try
	{
	"In Try/catch block"
	}
	catch
	{
	"Got exception"
	}
	finally
	{
	"Release resources"
	}```

Verwenden Sie die automatischen PowerShell Variablen, um auf Informationen über Ausnahmen zuzugreifen.	```try
	{
	Get-WFALogger -Info -message
	$("Creating
	Ipspace: " + $Ipspace)
	New-NaNetIpspace-Name $Ipspace
	}
	catch
	{
	Throw "Failed to create Ipspace.
	Message:
	" + $_.Exception.Message;
	}```

Perl-Fehlerbehandlung

Richtlinien	Beispiel

Perl umfasst keine native Sprachunterstützung für Try/Catch Blocks. Verwenden Sie Evaluierungsblöcke zum Prüfen und behandeln von Fehlern. Halten Sie Evaluierungsblöcke so klein wie möglich.	`eval {`
	
	`$wfa_util->sendLog('INFO',`
	`"Quiescing the relationship :`
	`$DestinationCluster://$Destination`
	`Vserver`
	`/$DestinationVolume"`
	`);`
	`$server->snapmirror_quiesce(`
	`'destination-vserver' =>`
	`$DestinationVserver,`
	`'destination-volume' =>`
	`$DestinationVolume`
	`);`
	`$wfa_util->sendLog('INFO',`
	`'Quiesce operation`
	`started successfully.');`
	`};`
	
	`$wfa_util->checkEvalFailure(`
	`"Failed to quiesce the SnapMirror`
	`relationship`
	`$DestinationCluster://$Destination`
	`Vserver`
	`/$DestinationVolume",`
	`$@`
	`);`

Allgemeine PowerShell und Perl Konventionen für WFA

Sie müssen bestimmte PowerShell- und Perl-Konventionen kennen, die in WFA zum Erstellen von Skripten verwendet werden, die mit vorhandenen Skripten identisch sind.

- Verwenden Sie Variablen, die Ihnen helfen, das zu klären, was das Skript tun soll.
- Lesbarer Code schreiben, der ohne Kommentare verstanden werden kann.
- Skripte und Befehle so einfach wie möglich halten.
- Für PowerShell Skripte:
  - Nutzen Sie nach Möglichkeit Commandlets.
  - Rufen Sie den .NET-Code auf, wenn kein Cmdlet verfügbar ist.
- Für Perl-Skripte:

- Beenden Sie immer „`die`“-Aussagen mit Newline-Zeichen.

  Wenn kein newline-Zeichen vorhanden ist, wird die Skriptliniennummer gedruckt, was für das Debuggen von Perl-Befehlen, die von WFA ausgeführt werden, nicht nützlich ist.

- Machen Sie im Modul „`getopt`“ die Zeichenfolgenargumente zu einem Befehl erforderlich.

Perl-Module mit Windows gebündelt

Einige Perl-Module werden mit der Windows Active State Perl Distribution for OnCommand Workflow Automation (WFA) gebündelt. Sie können diese Perl-Module in Ihrem Perl-Code zum Schreiben von Befehlen verwenden, nur wenn sie mit Windows gebündelt sind.

In der folgenden Tabelle sind die Perl-Datenbankmodule aufgeführt, die mit Windows für WFA gebündelt sind.

Datenbankmodul	Beschreibung
DBD::mysql	Perl5-Datenbank-Schnittstellentreiber, mit dem Sie eine Verbindung zur MySQL-Datenbank herstellen können.
Versuchen Sie::Winzig	Minimiert häufige Fehler durch Evaluierungsblöcke.
XML::LibXML	Schnittstelle zu libxml2, die XML- und HTML-Parser mit DOM-, SAX- und XMLReader-Schnittstellen bereitstellt.
DBD::Cassandra	Perl5-Datenbanktreiber für Cassandra, der die CQL3-Abfragesprache verwendet.

Überlegungen beim Hinzufügen benutzerdefinierter PowerShell und Perl Module

Beachten Sie bestimmte Überlegungen, bevor Sie OnCommand Workflow Automation (WFA) benutzerdefinierte PowerShell und Perl-Module hinzufügen. Benutzerdefinierte PowerShell- und Perl-Module ermöglichen die Verwendung benutzerdefinierter Befehle zum Erstellen von Workflows.

- Während der Ausführung von WFA Befehlen werden alle benutzerdefinierten PowerShell Module zum WFA Installationsverzeichnis hinzugefügt `/Posh/modules` Automatisch importiert.
- Alle benutzerdefinierten Perl-Module, die dem hinzugefügt wurden `WFA/perl` Das Verzeichnis ist in der Bibliothek *@Inc* enthalten.
- Individuelle PowerShell und Perl Module werden nicht als Teil des WFA Backups gesichert.
- Benutzerdefinierte PowerShell und Perl Module werden im Rahmen der WFA Restore-Operation nicht wiederhergestellt.

Sie müssen benutzerdefinierte PowerShell und Perl Module manuell sichern, um sie in eine neue WFA Installation zu kopieren.

Der Ordnername im Modulverzeichnis muss mit dem des Modulnamens übereinstimmen.

WFA Commandlets und Funktionen

OnCommand Workflow Automation (WFA) umfasst mehrere PowerShell Commandlets sowie PowerShell- und Perl-Funktionen, die Sie in Ihren WFA Befehlen nutzen können.

Mithilfe der folgenden PowerShell Befehle können Sie alle vom WFA Server bereitgestellten PowerShell Commandlets und Funktionen anzeigen:

- `Get-Command -Module WFAWrapper`
- `Get-Command -Module WFA`

Sie können alle Perl-Funktionen, die der WFA-Server zur Verfügung stellt, in anzeigen `WFAUtil.pm` Modul: In den Hilfebereichen, der Hilfe zu WFA PowerShell cmdlets und Hilfe zu WFA Perl Methoden ermöglicht das WFA Hilfemodul Support Links den Zugriff auf alle PowerShell cmdlets und Funktionen sowie auf die Perl Funktionen.

PowerShell und Perl WFA Module

Um Skripte für Ihre Workflows zu schreiben, müssen Sie die PowerShell oder Perl Module for OnCommand Workflow Automation (WFA) kennen.

PowerShell Module

Richtlinien	Beispiel
Verwenden Sie das Data ONTAP PS Toolkit, um APIs aufzurufen, sobald das Toolkit verfügbar ist.	Der `Add VLAN` Befehl verwendet das Toolkit wie folgt: `Add-NaNetVlan-Interface $Interface-Vlans$VlanID`
Wenn im Data ONTAP PS Toolkit keine Cmdlets verfügbar sind, verwenden Sie das `Invoke-SSH` Befehl zum Aufrufen der CLI auf Data ONTAP.	`Invoke-NaSsh-Name $ArrayName-Command "ifconfig -a"-Credential $Credentials`

Perl-Module

Das NaServer-Modul wird in WFA Befehlen verwendet. Das NaServer-Modul ermöglicht den Aufruf von Data ONTAP-APIs, die im aktiven Management von Data ONTAP-Systemen verwendet werden.

Richtlinien	Beispiel

Verwenden Sie das NaServer-Modul, um APIs aufzurufen, wann immer das NetApp Manageability SDK verfügbar ist.	Das folgende Beispiel zeigt, wie das NaServer-Modul für einen Vorgang zur Wiederaufnahme von SnapMirror verwendet wird:
	`eval {`
	`$wfa_util->sendLog('INFO', "Connecting to the cluster: $DestinationCluster" );`
	`my $server = $wfa_util->connect($DestinationClusterIp, $DestinationVserver);`
	`my $sm_info = $server->snapmirror_get( 'destination-vserver' => $DestinationVserver, 'destination-volume' => $DestinationVolume );`
	`my $sm_state = $sm_info->{'attributes'}->{'snapmirror-info'}->{'mirror-state'};`
	`my $sm_status = $sm_info->{'attributes'}->{'snapmirror-info'}->{'relationship-status'};`
	`$wfa_util->sendLog('INFO', "SnapMirror relationship is $sm_state ($sm_status)");`
	`if ($sm_status ne 'quiesced') {`
	`$wfa_util->sendLog('INFO', 'The status needs to be quiesced to resume transfer.');`
	`} else {`
	`my $result = $server->snapmirror_resume( 'destination-vserver' => $DestinationVserver, 'destination-volume' => $DestinationVolume );`
	`$wfa_util->sendLog('INFO',`

Richtlinien	Beispiel
Wenn keine Data ONTAP-API verfügbar ist, rufen Sie die Data ONTAP-CLI unter Verwendung der Methode des ausführbaren SystemClifdienstprogramms auf. ExecutSystemCli.1 wird nicht unterstützt und ist derzeit nur für Data ONTAP im 7-Modus verfügbar.	Keine

Überlegungen beim Konvertieren von PowerShell-Befehlen in Perl

Bei der Konvertierung von PowerShell Befehlen in Perl müssen Sie bestimmte wichtige Überlegungen beachten, da PowerShell und Perl über verschiedene Funktionen verfügen.

Eingabearten für Befehle

OnCommand Workflow Automation (WFA) ermöglicht Workflow-Designern bei der Definition eines Befehls die Verwendung von Arrays und Hash als Eingabe für den Befehl. Diese Eingabetypen können nicht verwendet werden, wenn der Befehl über Perl definiert ist. Wenn ein Perl-Befehl Array- und Hash-Eingaben akzeptieren soll, können Sie die Eingabe als Zeichenfolge im Designer definieren. Die Befehlsdefinition kann dann die Eingabe analysieren, die nach Bedarf an die Erstellung eines Arrays oder Hash übergeben wird. Die Beschreibung der Eingabe beschreibt das Format, in dem die Eingabe erwartet wird.

```
my @input_as_array  = split(',', $InputString); #Parse the input string of
format val1,val2 into an array

my %input_as_hash = split /[;=]/, $InputString; #Parse the input string of
format key1=val1;key2=val2 into a hash.
```

PowerShell Aussage

Die folgenden Beispiele zeigen, wie eine Array-Eingabe an PowerShell und Perl übergeben werden kann. Die Beispiele beschreiben den Input CronMonth, der den Monat angibt, in dem der Cron-Job ausgeführt werden soll. Die gültigen Werte sind ganze Zahlen -1 bis 11. Ein Wert von -1 zeigt an, dass der Zeitplan jeden Monat ausgeführt wird. Jeder andere Wert bezeichnet einen bestimmten Monat, wobei 0 Januar und 11 Dezember ist.

```
[parameter(Mandatory=$false, HelpMessage="Months in which the schedule
executes. This is a comma separated list of values from 0 through 11.
Value -1 means all months.")]
  [ValidateRange(-1, 11)]
  [array]$CronMonths,
```

Perl-Anweisung

```
GetOptions(
    "Cluster=s"         => \$Cluster,
    "ScheduleName=s"    => \$ScheduleName,
    "Type=s"            => \$Type,
    "CronMonths=s"      => \$CronMonths,
) or die 'Illegal command parameters\n';

sub get_cron_months {
    return get_cron_input_hash('CronMonths', $CronMonths, 'cron-month',
-1,
        11);
}

sub get_cron_input_hash {
    my $input_name   = shift;
    my $input_value  = shift;
    my $zapi_element = shift;
    my $low          = shift;
    my $high         = shift;
    my $exclude      = shift;

    if (!defined $input_value) {
        return undef;
    }

    my @values = split(',', $input_value);

    foreach my $val (@values) {
        if ($val !~ /^[+-]?\d+$/) {
            die
                "Invalid value '$input_value' for $input_name: $val must
be an integer.\n";
        }
        if ($val < $low || $val > $high) {
            die
                "Invalid value '$input_value' for $input_name: $val must
be from $low to $high.\n";
        }
        if (defined $exclude && $val == $exclude) {
            die
                "Invalid value '$input_value' for $input_name: $val is not
valid.\n";
        }
    }
    # do something
}
```

Befehlsdefinition

Ein einzeilter Ausdruck in PowerShell, der einen Pipe Operator verwendet, muss in Perl in mehrere Blöcke von Anweisungen erweitert werden, um dieselbe Funktionalität zu erreichen. Die folgende Tabelle enthält ein Beispiel für einen der Wartebefehle.

PowerShell Aussage	Perl-Anweisung
`# Get the latest job which moves the specified volume to the specified aggregate.` `$job = Get-NcJob -Query $query`	where {$_.JobDescription -eq "Split" + $VolumeCloneName}

Select-Object -First 1 ----	`my $result = $server->job_get_iter(`
	`'query' => {'job-type' => 'VOL_CLONE_SPLIT'},`
	`'desired-attributes' => {`
	`'job-type' => '',`
	`'job-description' => '',`
	`'job-progress' => '',`
	`'job-state' => ''`
	`}`
	`);`
	`my @jobarray;`
	`for my $job (@{ $result->{'attributes-list'}})`
	`{`
	`my $description = $job->{'job-description'};`
	`if($description =~ /$VolumeCloneName/)`
	`{`
	`push(@jobarray, $job)`
	`}`
	`}`

Richtlinien für WFA Bausteine

Beachten Sie unbedingt die Richtlinien zur Nutzung der Workflow-Automatisierungs-Bausteine.

Richtlinien für SQL in WFA

Sie müssen die Richtlinien zur Verwendung von SQL in OnCommand Workflow Automation (WFA) kennen, um SQL-Abfragen für WFA zu schreiben.

SQL wird an folgenden Stellen in WFA verwendet:

- SQL-Abfragen zum Befüllen der Benutzereingaben zur Auswahl
- SQL-Abfragen zum Erstellen von Filtern zum Filtern von Objekten eines bestimmten Wörterbucheingabetyps
- Statische Daten in Tabellen in der Spielplatzdatenbank
- Ein benutzerdefinierter Quelltyp des SQL-Typs, bei dem die Daten aus einer externen Datenquelle extrahiert werden müssen, z. B. aus einer benutzerdefinierten Configuration Management Database (CMDB).
- SQL fragt nach Reservierungs- und Verifikationsskripten ab

Richtlinien	Beispiel
SQL-reservierte Schlüsselwörter müssen in Großbuchstaben enthalten sein.	`SELECT vserver.name FROM cm_storage.vserver vserver`
Tabelle- und Spaltennamen müssen in Kleinbuchstaben enthalten sein.	Tabelle: Aggregat Spalte: Used_space_mb
Trennen Sie Wörter mit einem Unterstrich (_) Zeichen. Leerzeichen sind nicht zulässig.	Array_Performance
Tabellenname wird in Singular definiert. Eine Tabelle ist eine Sammlung von einem oder mehreren Einträgen.	„`Funktion`“, nicht „`Funktionen`“

Richtlinien	Beispiel

Verwenden Sie in AUSGEWÄHLTEN Abfragen Tabellenaliase mit aussagekräftigen Namen.	```
	SELECT
	    vserver.name
	FROM
	    cm_storage.cluster cluster,
	    cm_storage.vserver vserver
	WHERE
	    vserver.cluster_id =
	cluster.id
	    AND cluster.name =
	'${ClusterName}'
	    AND vserver.type = 'cluster'
	ORDER BY
	    vserver.name ASC
	``

Richtlinien	Beispiel

Wenn Sie in einer Filterabfrage oder Benutzerabfrage auf einen Filtereingabeparameter oder Benutzereingabeparameter verweisen müssen, verwenden Sie die Syntax als '€{inputVariableName}.Sie können die Syntax auch verwenden, um in Reservierungsskripten und Verifikationsskripten auf einen Parameter der Befehlsdefinition zu verweisen.	`SELECT`
	`volume.name AS Name,`
	`aggregate.name as Aggregate,`
	`volume.size_mb AS 'Total Size (MB)',`
	`voulme.used_size_mb AS 'Used Size (MB)',`
	`volume.space_guarantee AS 'Space Guarantee'`
	`FROM`
	`cm_storage.cluster,`
	`cm_storage.aggregate,`
	`cm_storage.vserver,`
	`cm_storage.volume`
	`WHERE`
	`cluster.id = vserver.cluster_id`
	`AND aggregate.id = volume.aggregate_id`
	`AND vserver.id = voulme.vserver_id`
	`AND vserver.name = '${VserverName}'`
	`AND cluster.name = '${ClusterName}'`
	`ORDER BY`
	`volume.name ASC`

Verwenden Sie Kommentare für komplexe Abfragen. Einige der unterstützten Kommentarstile in Abfragen sind wie folgt:
• „--“ bis zum Ende der Zeile
Nach dem zweiten Bindestrich in diesem Kommentarstil ist ein Leerzeichen erforderlich.
• Von einem “#” Zeichen bis zum Ende der Zeile
• Von einem „/`**" to the following "**/`„Sequenz
	`/*`
	`multi-line`
	`comment`
	`*/`
	`--line comment`
	`SELECT`
	`ip as ip, # comment till end of this line`
	`NAME as name`
	`FROM --end of line comment`
	`storage.array`

Richtlinien für Funktionen von WFA

Sie können Funktionen erstellen, um häufig verwendete und komplexere Logik in einer benannten Funktion einzukapseln und die Funktion dann als Befehlsparameter-Werte oder Filterparameter-Werte in OnCommand Workflow Automation (WFA) wiederzuverwenden.

Richtlinien	Beispiel
Verwenden Sie Camel Case für einen Funktionsnamen.	KalkulationVolumeGröße
Variablennamen sollten in einfachem Englisch sein und sich auf die Funktionalität der Funktion beziehen.	Trennungszeichen ByDelimiter
Verwenden Sie keine Abkürzungen.	KalküteVolumeSize, *Not* calcVolSize
Funktionen werden mit MVFLEX Expression Language (MVEL) definiert.	Keine
Die Funktionsdefinition sollte nach den offiziellen Java Programmiersprachen Richtlinien angegeben werden.	Keine

Richtlinien für Einträge im WFA Wörterbuch

Beim Erstellen von Wörterbucheinträgen in OnCommand Workflow Automation (WFA) müssen Sie die Richtlinien kennen.

Richtlinien	Beispiel
Namen von Wörterbucheintragsnamen dürfen nur alphanumerische Zeichen und Unterstriche enthalten.	Cluster_Lizenz Switch_23
Die Namen der Wörterbucheingabe müssen mit einem Großbuchstaben beginnen. Beginnen Sie jedes Wort im Namen mit einem Großbuchstaben und trennen Sie jedes Wort mit einem Unterstrich (_).	Datenmenge Array_Lizenz
Attributnamen für Wörterbucheintrag sollten den Namen des Wörterbucheintrags nicht enthalten.	Keine
Attribute und Verweise in einem Wörterbucheintrag müssen in Kleinbuchstaben enthalten sein.	Aggregat, Größe_mb
Trennen Sie Wörter mit einem Unterstrich. Leerzeichen sind nicht zulässig.	Ressourcen-Pool

Richtlinien	Beispiel
Wörterbucheinträge können keine Referenzen enthalten, die von einem anderen Schema stammen. Wenn ein Wörterbucheintrag einen Querverweis auf ein Objekt in einem anderen Schema erfordert, stellen Sie sicher, dass alle natürlichen Schlüssel des Objekts, auf das verwiesen wird, im Wörterbucheintrag vorhanden sind.	Array_Performance-Wörterbuch-Eintrag erfordert alle natürlichen Schlüssel des Array-Wörterbuchs als direkte Attribute darin.
Verwenden Sie die entsprechenden Datentypen für Attribute.	Keine
Verwenden Sie den langen Datentyp für die Größe oder die speicherbezogenen Attribute.	Size_mb und Available_size_mb im Storage.Volume-Wörterbuch-Eintrag
Verwenden Sie Enum, wenn ein Attribut einen festen Satz von Werten hat.	raid_type in Storage.Volume-Wörterbuch-Eintrag
Legen Sie „`to be gecached`“ für ein Attribut oder eine Referenz fest, wenn eine Datenquelle Wert für dieses Attribut oder diesen Verweis gibt.für die Active IQ Unified Manager Datenquelle fügen Sie Cache-fähige Attribute hinzu, wenn die Datenquelle den Wert für sie liefern kann.	Keine
Set „`kann Null`“ als wahr sein, wenn die Datenquelle, die den Wert für dieses Attribut oder diesen Verweis bereitstellt, Null zurückgeben kann.	Keine
Geben Sie jedem Attribut und jeder Referenz eine aussagekräftige Beschreibung an.die Beschreibung wird bei der Gestaltung eines Workflows in Befehlsdetails angezeigt.	Keine
Verwenden Sie „`id`“ nicht als Namen eines Attributs in Glossareinträgen. Es ist für die interne Verwendung von WFA reserviert.	Keine

Verwandte Informationen

Verweise auf Lernmaterial

Richtlinien für Befehle

Zum Erstellen von Befehlen in OnCommand Workflow Automation (WFA) müssen Sie die Richtlinien kennen.

Richtlinien	Beispiel
Verwenden Sie einen leicht identifizierbaren Namen für Befehle.	`Create Qtree`
Verwenden Sie Leerzeichen, um Wörter zu begrenzen, und jedes Wort muss mit einem Großbuchstaben beginnen.	`Create Volume`
Geben Sie eine Beschreibung zur Erläuterung der Funktionalität des Befehls an, einschließlich des erwarteten Ergebnisses der optionalen Parameter.	Keine
Standardmäßig beträgt die Zeitüberschreitung für Standardbefehle 600 Sekunden. Die Standard-Zeitüberschreitung wird beim Erstellen des Befehls festgelegt. Ändern Sie den Standardwert nur, wenn der Befehl möglicherweise länger dauert.	`Create Volume` Befehl
Erstellen Sie bei langlebigen Vorgängen zwei Befehle: Einen, um den lang ausgeführten Vorgang aufzurufen, und einen anderen, um den Fortschritt des Vorgangs regelmäßig zu melden. Der erste Befehl sollte ein sein `Standard Execution` Befehlstyp und die zweite sollte sein `Wait for Condition` Befehlstyp.	`Create VSM` Und `Wait for VSM` Befehle
Setzen Sie das ein `Wait for condition` Befehlsnamen mit „`wait`“ zur einfachen Identifizierung.	`Wait for CM Volume Move`
Verwenden Sie ein geeignetes Wartungsintervall für die Befehle „`Wait for Condition`“. Der angegebene Wert bestimmt das Intervall, in dem der Abfragebefehl ausgeführt wird, um zu prüfen, ob der lang laufende Vorgang abgeschlossen ist.	Abtastintervall 60er für das `Wait for VSM` Befehl
Für das `Wait for condition` Verwenden Sie Befehle eine angemessene Zeitüberschreitung auf der Grundlage der erwarteten Zeit, während der lange laufende Vorgang abgeschlossen wird. Die erwartete Zeit kann erheblich länger sein, wenn der Vorgang die Datenübertragung über ein Netzwerk umfasst.	Ein VSM-Basistransfer kann viele Tage in Anspruch nehmen. Daher beträgt die angegebene Zeitüberschreitung 6 Tage.

Zeichenfolgendarstellung

Die Zeichenfolgendarstellung für einen Befehl zeigt die Details eines Befehls in einem Workflow-Design während der Planung und Ausführung an. In der String-Darstellung für einen Befehl können nur die Befehlsparameter verwendet werden.

Richtlinien	Beispiel
Vermeiden Sie die Verwendung von Attributen, die keinen Wert haben. Ein Attribut ohne Wert wird als NA angezeigt.	VolName 10.68.66.212[NA]aggr1/testVol7
Trennen Sie verschiedene Einträge in der String-Darstellung mit den folgenden Trennzeichen: [ ] , / :	*ArrayName[ArrayIp]*
Geben Sie jedem Wert in der Zeichenfolgendarstellung aussagekräftige Beschriftungen an.	*Volume name=VolumeName*

Sprache der Befehlsdefinition

Befehle können mithilfe der folgenden unterstützten Skriptsprachen geschrieben werden:

- PowerShell
- Perl

Definition von Befehlsparametern

Die Befehlsparameter werden mit Name, Beschreibung, Typ und einem Standardwert für den Parameter beschrieben und ob der Parameter obligatorisch ist. Der Parametertyp kann String, Boolean, Integer, Long, Double, sein Enum, DateTime, Capacity, Array, Hashtable, Kennwort oder XmlDocument. Während die Werte für die meisten Typen intuitiv sind, sollten die Werte für Array und Hashtable in einem bestimmten Format vorliegen, wie in der folgenden Tabelle beschrieben:

Richtlinien	Beispiel
Stellen Sie sicher, dass der Wert für einen Array-Eingabetyp eine Liste von Werten ist, die durch Komma getrennt sind.	`[parameter(Mandatory=$false, HelpMessage="Months in which the schedule executes.")] [array]$CronMonths` Eingabe wird wie folgt übergeben: 0,3,6,9
Stellen Sie sicher, dass der Wert für einen Hashtable-Eingabetyp eine Liste von Key=value pairs ist, getrennt durch Semikolon.	`[parameter(Mandatory=$false, HelpMessage="Volume names and size (in MB)")] [hashtable]$VolumeNamesAndSize` Eingabe wird wie folgt übergeben: Volume1=100;Volume2=250;Volume3=50

Richtlinien für Workflows

Sie müssen die Richtlinien zum Erstellen oder Ändern eines vordefinierten Workflows für OnCommand Workflow Automation (WFA) kennen.

Allgemeine Richtlinien

Richtlinien	Beispiel
Benennen Sie den Workflow, sodass er den Vorgang wiedergibt, der vom Storage Operator ausgeführt wird.	`Create a CIFS Share`
Für Workflow-Namen, setzen Sie den Anfangsbuchstaben des ersten Wortes und jedes Wort, das ein Objekt ist. Buchstaben für Abkürzungen und Akronyme schreiben.	Datenmenge Qtree Erstellen Sie eine Clustered Data ONTAP Qtree CIFS Share
Fügen Sie bei Workflow-Beschreibungen alle wichtigen Schritte des Workflows ein, einschließlich aller Voraussetzungen, Ergebnisse des Workflows oder bedingter Aspekte der Ausführung.	Siehe Beschreibung des Beispielworkflows `Create VMware NFS Datastore on Clustered Data ONTAP Storage`, Dazu gehören die Voraussetzungen.
Setzen Sie „`bereit für die Produktion`“ auf `true` Nur wenn der Workflow für die Produktion bereit ist und auf der Portalseite angezeigt werden kann.	Keine
Setzen Sie standardmäßig „`chaue reserved Elements`“ auf true. Bei der Vorschau eines Workflows für die Ausführung berücksichtigt der WFA Planner alle Objekte, die zusammen mit den bestehenden Objekten in der Cache-Datenbank reserviert sind. Wenn diese Option auf festgelegt ist, werden die Auswirkungen anderer parallel ausgeführter geplanter Workflows oder Arbeitsschritte bei der Planung eines bestimmten Workflows berücksichtigt `true`.	• Szenario 1 Workflow 1 erstellt ein Volume und wird für eine Woche später ausgeführt. Workflow 2 erstellt qtrees oder LUNs in Volumes, nach denen gesucht wird. Falls Workflow 2 innerhalb eines Tages oder so ausgeführt wird, sollten Sie für Workflow 2 „`chaltest reservierte Elemente`“ deaktivieren, um zu verhindern, dass das Volume, das innerhalb einer Woche erstellt werden soll, berücksichtigt wird. • Szenario 2 Workflow 1 verwendet den `Create Volume` Befehl. Wenn ein geplanter Workflow 2 100 GB in einem Aggregat verbraucht, muss Workflow 1 die Anforderungen für Workflow 2 während der Planung berücksichtigen.

Richtlinien	Beispiel
Standardmäßig ist „`Enable Element Existenzvalidierung`“ auf festgelegt `true`.	• Szenario 1 Wenn Sie einen Workflow erstellen, der zuerst ein Volume mit dem Namen entfernt, verwenden Sie den Befehl `Remove Volume` Nur wenn das Volume vorhanden ist und das Volume mit einem anderen Befehl wie z. B. neu erstellt wird `Create Volume` Oder `Clone Volume`, Dann sollte der Workflow dieses Flag nicht verwenden. Der Effekt des Entfernens des Volumens steht dem nicht zur Verfügung `Create volume` Befehl, wodurch der Workflow fehlschlagen wird. • Szenario 2 Der `Create Volume` Befehl wird in einem Workflow mit einem bestimmten Namen als „`vol198`“ verwendet. Wenn diese Option auf „true“ gesetzt ist, überprüft WFA Planner bei der Planung, ob ein Volume mit diesem Namen im angegebenen Array vorhanden ist. Wenn das Volume vorhanden ist, schlägt der Workflow während der Planung fehl.
Wenn derselbe Befehl mehr als einmal in einem Workflow ausgewählt ist, geben Sie entsprechende Anzeigenamen für die Befehlsinstanzen an.	Im Beispiel-Workflow „`Erstellen, Zuordnen und Schützen von LUNs mit SnapVault`“ wird der verwendet `Create Volume` Zweimal Befehl. Die Anzeigenamen werden jedoch als verwendet `Create Primary Volume` Und `Create Secondary Volume` Entsprechend für das primäre Volume und das gespiegelte Ziel-Volume.

Benutzereingaben

Richtlinien	Beispiel

Namen:
 	
• Starten Sie den Namen mit dem Zeichen „` €`“.
• Verwenden Sie einen Großbuchstaben am Anfang jedes Wortes.
• Verwenden Sie Großbuchstaben für alle Begriffe und Abkürzungen.
• Keine Unterstriche verwenden.
	`$Array`
	
	`$VolumeName`

Richtlinien	Beispiel

Namen anzeigen:
• Verwenden Sie einen Großbuchstaben am Anfang jedes Wortes.
• Trennen Sie Wörter mit Leerzeichen.
• Wenn Eingänge bestimmte Einheiten haben, geben Sie die Einheit in Klammern im Anzeigenamen direkt an.
	`Volume Name`
	
	`Volume Size (MB)`

Beschreibungen: • Geben Sie für jede Benutzereingabe eine aussagekräftige Beschreibung an. • Stellen Sie bei Bedarf Beispiele bereit. Dies sollte insbesondere dann erfolgen, wenn die Benutzereingaben in einem bestimmten Format vorliegen sollen. Die Benutzereingabebeschreibungen werden als Tooltips für die Benutzereingaben bei der Workflow-Ausführung angezeigt.	Initiatoren, die zu einer „`Initiatorgruppe`“ hinzugefügt werden sollen. Beispielsweise IQN oder WWPN des Initiators.
Typ: Wählen Sie als Typ Enum aus, wenn Sie die Eingabe auf einen bestimmten Satz von Werten beschränken möchten.	Protokoll: „`iscsi`“, „`fcp`“, „`mixed`“

Typ: Wählen Sie Query als Typ aus, wenn der Benutzer aus Werten auswählen kann, die im WFA Cache verfügbar sind.	Array USD: ABFRAGETYP mit Abfrage wie folgt:
	
	``
	SELECT
	    ip, name
	FROM
	    storage.array
	``

Typ: Markieren Sie die Benutzereingabe als gesperrt, wenn die Benutzereingabe auf die Werte beschränkt werden soll, die von einer Abfrage erhalten werden oder nur auf die unterstützten Enum-Typen beschränkt sein sollten.	Array: Gesperrt Abfragetyp: Es können nur Arrays im Cache ausgewählt werden. €Protokoll: Gesperrter Enum-Typ mit gültigen Werten wie iscsi, fcp, gemischt. Kein anderer Wert als der gültige Wert wird unterstützt.
Typ: Abfrage-Typ zusätzliche Spalten als Rückgabewerte in der Abfrage hinzufügen, wenn es dem Speicherbetreiber hilft, die richtige Wahl der Benutzereingabe zu treffen.	EUR Aggregat: Geben Sie Name, Gesamtgröße, verfügbare Größe, so dass der Betreiber die Attribute kennt, bevor Sie das Aggregat auswählen.

Richtlinien	Beispiel

Typ: Abfrage TypeSQL Abfrage für Benutzereingaben kann auf alle anderen Benutzer-Eingaben vor ihm beziehen. Dadurch können die Ergebnisse einer Abfrage auf Basis anderer Benutzereingaben wie z. B. vFiler Einheiten eines Arrays, Volumes eines Aggregats, LUNs in einer Storage Virtual Machine (SVM) begrenzt werden.	Im Beispielworkflow `Create a Clustered Data ONTAP Volume`, Die Abfrage für VserverName lautet wie folgt:
	`SELECT`
	`vserver.name`
	`FROM`
	`cm_storage.cluster cluster,`
	`cm_storage.vserver vserver`
	`WHERE`
	`vserver.cluster_id = cluster.id`
	`AND cluster.name = '${ClusterName}'`
	`AND vserver.type = 'cluster'`
	`ORDER BY`
	`vserver.name ASC`
	Die Abfrage bezieht sich auf €{clusterName}, wobei USD clusterName der Name der Benutzereingaben vor der Benutzereingabe für VserverName ist.

Typ: Verwenden Sie Booleschen Typ mit Werten als “true, false” für Benutzereingaben, die boolesch sind. Dies hilft beim Schreiben interner Ausdrücke im Workflow-Design mit der Benutzereingabe direkt. Beispiel: €UserInputName statt €UserInputName == ''`Yes'.	`$CreateCIFSShare`: Boolescher Typ mit gültigen Werten als „`true`“ oder „`false`“
Typ:für String- und Zahlentyp verwenden Sie in der Spalte Werte reguläre Ausdrücke, wenn Sie den Wert mit bestimmten Formaten validieren möchten. Verwenden Sie regelmäßige Ausdrücke für IP-Adresse und Netzwerkmaskeneingaben.	Ortsspezifische Benutzereingaben können als „`[A-Z][A-Z]\-0[1-9]`“ angegeben werden. Diese Benutzereingabe akzeptiert Werte wie „`US-01`“, „`NB-02`“, nicht jedoch „`nb-00`“.
Typ: Für den Zahlentyp kann in der Spalte Werte eine Bereichsbasierte Validierung angegeben werden.	Für Anzahl der zu erstellenden LUNs ist der Eintrag in der Spalte Werte 1-20.
Gruppe: Gruppieren Sie die entsprechenden Benutzereingaben in den entsprechenden Buckets und benennen Sie die Gruppe.	„`sStorage Details`“ für alle Storage-bezogenen Benutzereingaben. „`DatStore Details`“ für alle Eingaben von Benutzern, die mit VMware zusammenhängen.

Richtlinien	Beispiel
Obligatorisch: Wenn der Wert einer Benutzereingabe für die Ausführung des Workflows erforderlich ist, markieren Sie die Benutzereingabe als obligatorisch. Dadurch wird sichergestellt, dass die Eingabe des Benutzers vom Benutzer akzeptiert wird.	„` Dollar VolumeName`“ im Workflow „`NFS-Volume erstellen`“.
Standardwert: Wenn eine Benutzereingabe einen Standardwert hat, der für die meisten Workflow-Ausführungen arbeiten kann, geben Sie die Werte an. Dadurch kann der Benutzer während der Ausführung weniger Eingaben zur Verfügung stellen, wenn der Standardwert dem Zweck dient.	Keine

Konstanten, Variablen und gibt Parameter zurück

Richtlinien	Beispiel
Konstanten: Definieren Sie Konstanten bei der Verwendung eines gemeinsamen Werts für die Definition von Parametern zu mehreren Befehlen.	*AGGREGATE_OVERCOMMITMENT_THRESHOLD* in `Create, map, and protect LUNs with SnapVault sample workflow.`
Konstanten:Namen • Verwenden Sie einen Großbuchstaben am Anfang jedes Wortes. • Verwenden Sie Großbuchstaben für alle Begriffe und Abkürzungen. • Keine Unterstriche verwenden. • Verwenden Sie Großbuchstaben für alle Buchstaben konstanter Namen.	*AGGREGATE_USED_SPACE_THRESHOLD* *ActuVolumeSizeInMB*
Variablen: Geben Sie einem Objekt einen Namen an, das in einer der Befehlsparameter-Felder definiert ist. Variablen sind automatisch generierte Namen und können geändert werden.	Keine
Variablen: Namen verwenden Kleinbuchstaben für Variablennamen.	volume1 cifs_Freigabe
Rückgabeparameter: Verwenden Sie Rückgabeparameter, wenn die Workflow-Planung und -Ausführung während der Planung einige berechnete oder ausgewählte Werte zurückgeben soll. Die Werte werden im Vorschaumodus verfügbar gemacht, wenn der Workflow auch von einem Webservice ausgeführt wird.	Aggregat: Wenn das Aggregat mit der Ressourcenauswahllogik ausgewählt wird, kann das tatsächlich ausgewählte Aggregat als Rückgabeparameter definiert werden.

Richtlinien zum Erstellen von Validierungsskripten für Remote-Systemtypen

Beachten Sie die Richtlinien zum Erstellen von Validierungsskripten, die zum Testen der in OnCommand Workflow Automation (WFA) definierten Remote-Systemtypen verwendet werden.

- Das von Ihnen erstellte Perl-Skript muss dem Beispielskript im Fenster Gültigkeitsskript ähnlich sein.
- Die Ausgabe Ihres Validierungsskripts muss dem des Beispielskripts ähnlich sein.

Beispiel für ein Validierungsskript

```
# Check connectivity.
# Return 1 on success.
# Return 0 on failure and set $message
sub checkCredentials {
my ($host, $user, $passwd, $protocol, $port, $timeout) = @_;
#
# Please add the code to check connectivity to $host using $protocol here.
#
return 1;
}
```

Richtlinien zum Erstellen von Datenquelltypen

Beachten Sie die Richtlinien zum Erstellen von Datenquelltypen, die zum Definieren benutzerdefinierter Datenquellen für OnCommand Workflow Automation (WFA) verwendet werden.

Sie können einen Datenquelltyp mit einer der folgenden Methoden definieren:

- SQL: Sie können mithilfe der WFA SQL Richtlinien Abfragen aus Datenquellen definieren, die auf einer externen Datenbank basieren.
- SKRIPT: Sie können ein PowerShell-Skript schreiben, das die Daten für ein bestimmtes Schema von Wörterbucheinträgen bereitstellt.

Die Richtlinien zum Erstellen von Datenquelltypen sind wie folgt:

- Sie sollten PowerShell-Sprache verwenden, um ein Skript zu erstellen.
- Das PowerShell-Skript sollte die Ausgabe für jeden Glossareintrag in seinem aktuellen Arbeitsverzeichnis bereitstellen.
- Die Datendateien sollten benannt werden `dictionary_entry.csv`, Der Name des Wörterbucheintrags sollte in Kleinbuchstaben enthalten sein.

  Der vordefinierte Quelltyp der Daten, der Informationen von Performance Advisor erfasst, verwendet einen SKRIPTBASIERTEN Datenquelltyp. Die Ausgabedateien werden benannt `array_performance.csv` Und `aggregate_performance.csv`.

- Der `.csv` Die Datei sollte den Inhalt in der genauen Reihenfolge wie die Attribute des Wörterbucheintrags enthalten.

  Ein Eintrag aus dem Wörterbuch enthält Attribute in der folgenden Reihenfolge: Array_ip, Datum, Tag, Stunde, cpu_beschäftigt, Total_OPS_per_sec, Disk_Throughput_per_sec

  Das PowerShell Skript fügt dem Daten hinzu `.csv` Datei in derselben Reihenfolge.

```
$values = get-Array-CounterValueString ([REF]$data)
Add-Content $arrayFile ([byte[]][char[]] "\N
t$arrayIP't$date't$day't$hour't$values'n")
```

- Sie sollten Encoding verwenden, um sicherzustellen, dass die Datenausgabe aus dem Skript exakt in den WFA Cache geladen ist.
- Sie sollten \N verwenden, wenn Sie einen Null-Wert in das eingeben `.csv` Datei: