Linee guida per la codifica per WFA

OnCommand Workflow Automation

NetApp
October 09, 2025

Sommario

Linee guida per la codifica per WFA

È necessario comprendere le linee guida generali per la codifica WFA (OnCommand Workflow Automation), le convenzioni di denominazione e i consigli per la creazione di vari blocchi di base come filtri, funzioni, comandi e flussi di lavoro.

Linee guida per le variabili

Quando si crea un comando o un tipo di origine dati, è necessario conoscere le linee guida per le variabili PowerShell e Perl in OnCommand Workflow Automation (Wfa).

Variabili PowerShell

Linee guida	Esempio
Per i parametri di input dello script: • USA il caso Pascal. • Non utilizzare i caratteri di sottolineatura. • Non utilizzare abbreviazioni.	`$VolumeName` `$AutoDeleteOptions` `$Size`
Per le variabili interne dello script: • USA la custodia Camel. • Non utilizzare i caratteri di sottolineatura. • Non utilizzare abbreviazioni.	`$newVolume` `$qtreeName` `$time`
Per le funzioni: • USA il caso Pascal. • Non utilizzare i caratteri di sottolineatura. • Non utilizzare abbreviazioni.	`GetVolumeSize`
I nomi delle variabili non sono sensibili al maiuscolo/minuscolo. Tuttavia, per migliorare la leggibilità, non utilizzare maiuscole diverse per lo stesso nome.	`$variable` è uguale a. `$Variable.`
I nomi delle variabili devono essere in inglese e devono essere correlati alla funzionalità dello script.	Utilizzare `$name` e non `$a.`
Dichiarare esplicitamente il tipo di dati per ogni variabile.	nome [stringa] [int]dimensione

Linee guida	Esempio
Non utilizzare caratteri speciali (! @ N. e % , .) e spazi.	Nessuno
Non utilizzare parole chiave riservate di PowerShell.	Nessuno
Raggruppare i parametri di input inserendo prima i parametri obbligatori seguiti dai parametri facoltativi.	`param( [parameter(Mandatory=$true)] [string]$Type,  [parameter(Mandatory=$true)] [string]$Ip,  [parameter(Mandatory=$false)] [string]$VolumeName ) `
Commentare tutte le variabili di input utilizzando *`HelpMessage`* annotazione con un messaggio di aiuto significativo.	`[parameter(Mandatory=$false,HelpMessage="LUN to map")] [string]$LUNName`
Non utilizzare “Filer” come nome di variabile; utilizzare invece “Array”.	Nessuno
Utilizzare *`ValidateSet`* annotazione nei casi in cui l’argomento ottiene valori enumerati. Questo si traduce automaticamente in tipo di dati Enum per il parametro.	`[parameter(Mandatory=$false,HelpMessage="Volume state")] [ValidateSet("online","offline","restricted")] [string]$State`
Aggiungere un alias a un parametro che termina con “_Capacity” per indicare che il parametro è del tipo di capacità.	Il comando “Create Volume” utilizza gli alias come segue: `[parameter(Mandatory=$false,HelpMessage="Volume increment size in MB")] [Alias("AutosizeIncrementSize_Capacity")] [int]$AutosizeIncrementSize`

Linee guida	Esempio
Aggiungere un alias a un parametro che termina con "_Password" per indicare che il parametro è di tipo password.	``` 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) ```

Variabili Perl

Linee guida	Esempio
Per i parametri di input dello script: • USA il caso Pascal. • Non utilizzare i caratteri di sottolineatura. • Non utilizzare abbreviazioni.	`$VolumeName` `$AutoDeleteOptions` `$Size`
Non utilizzare abbreviazioni per le variabili interne dello script.	`$new_volume` `$qtree_name` `$time`
Non utilizzare abbreviazioni per le funzioni.	`get_volume_size`
I nomi delle variabili sono sensibili al maiuscolo/minuscolo. Per migliorare la leggibilità, non utilizzare maiuscole diverse per lo stesso nome.	`$variable` non è uguale a. `$Variable.`
I nomi delle variabili devono essere in inglese e devono essere correlati alla funzionalità dello script.	Utilizzare `$name` e non `$a.`
Raggruppare i parametri di input inserendo per primi i parametri obbligatori, seguiti dai parametri facoltativi.	Nessuno

Linee guida	Esempio
Nella funzione GetOptions, dichiarare esplicitamente il tipo di dati di ciascuna variabile per i parametri di input.	`GetOptions( "Name=s"=>\$Name, "Size=i"=>\$Size )`
Non utilizzare “Filer” come nome di variabile; utilizzare invece “Array”.	Nessuno
Perl non include *`ValidateSet`* annotazione per i valori enumerati. Utilizzare istruzioni “if” esplicite per i casi in cui l’argomento ottiene valori enumerati.	`if (defined$SpaceGuarantee&&!($SpaceGuaranteeeq'none'`
	$SpaceGuaranteeeq’volume'
	$SpaceGuaranteeeq’file')) { die’Illegal SpaceGuarantee argument: \".$SpaceGuarantee.'\"; } ----
Tutti i comandi Perl WFA devono utilizzare il pragma “strit” per scoraggiare l’utilizzo di costrutti non sicuri per variabili, riferimenti e subroutine.	`use strict; # the above is equivalent to use strictvars; use strictsubs; use strictrefs;`
Tutti i comandi Perl WFA devono utilizzare i seguenti moduli Perl: • Getopt Viene utilizzato per specificare i parametri di input. • WFAUtil Viene utilizzato per le funzioni di utility fornite per la registrazione dei comandi, il reporting dell’avanzamento dei comandi, la connessione ai controller di array e così via.	`use Getopt::Long; use NaServer; use WFAUtil;`

Linee guida per il rientro

È necessario conoscere le linee guida per il rientro durante la scrittura di uno script PowerShell o Perl per OnCommand Workflow Automation (Wfa).

Linee guida	Esempio
Una scheda equivale a quattro spazi vuoti.	
Utilizzare le schede e le parentesi graffe per visualizzare l'inizio e la fine di un blocco.	Script PowerShell `if ($pair.length-ne 2) { throw "Got wrong input data" }` Script Perl `if (defined $MaxDirectorySize) { # convert from MBytes to Bytes my $MaxDirectorySizeBytes = $MaxDirectorySize * 1024 * 1024; }`
Aggiungere righe vuote tra set di operazioni o blocchi di codice.	`$options=$option.trim(); $pair=$option.split(" "); Get-WFAlogger -Info -messages $("split options: "+ $Pair)`

Linee guida per i commenti

È necessario conoscere le linee guida per i commenti di PowerShell e Perl nei propri script per OnCommand Workflow Automation (Wfa).

Commenti su PowerShell

Linee guida	Esempio
Utilizzare il carattere n. per un commento a una riga.	`# Single line comment $options=$option.trim();`

Linee guida	Esempio
Utilizzare il carattere n. per un commento di fine riga.	`$options=$option.trim(); # End of line comment`
Utilizzare i caratteri < n. e n.> per un commento di blocco.	`<# This is a block comment #> $options=$option.trim();`

Commenti di Perl

Linee guida	Esempio
Utilizzare il carattere n. per il commento a riga singola.	`# convert from MBytes to Bytes my $MaxDirectorySizeBytes = $MaxDirectorySize * 1024 * 1024;`
Utilizzare il carattere n. per il commento di fine riga.	`my $MaxDirectorySizeBytes = $MaxDirect orySiZe * 1024 * 1024; # convert to Bytes`

Linee guida	Esempio
Utilizzare il carattere N. in ogni riga con un numero vuoto all'inizio e alla fine per creare un bordo di commento per i commenti su più righe.	```# # 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 #```
Non includere i codici commentati e inattivi nei comandi WFA. Tuttavia, a scopo di test, è possibile utilizzare il meccanismo POD (Plain Old Documentation) per commentare il codice.	```=begin comment # Set deduplication if(defined $Deduplication && $Deduplication eq "enabled") { $wfaUtil- >sendLog("Enabling Deduplication"); } =end comment =cut```

Linee guida per la registrazione

È necessario conoscere le linee guida per la registrazione quando si scrive uno script PowerShell o Perl per OnCommand Workflow Automation (Wfa).

Registrazione PowerShell

Linee guida	Esempio
Utilizzare il cmdlet Get-WFALogger per la registrazione.	```Get-WFALogger -Info -message "Creating volume"```

Linee guida	Esempio
Registra ogni azione che richiede l'interazione con pacchetti interni come Data ONTAP, VMware e PowerCLI. Tutti i messaggi di log sono disponibili nei registri di esecuzione nella cronologia dello stato di esecuzione dei flussi di lavoro.	Nessuno
Registrare tutti gli argomenti pertinenti che vengono passati ai pacchetti interni.	Nessuno
Utilizzare i livelli di registro appropriati quando si utilizza il cmdlet Get-WFALogger, a seconda del contesto di utilizzo. -Info, -Error, -WARN e -Debug sono i vari livelli di log disponibili. Se non viene specificato un livello di log, il livello di log predefinito è Debug.	Nessuno

Registrazione delle prestazioni

Linee guida	Esempio
Utilizzare WFAUtil sendLog per la registrazione.	`my wfa_util = WFAUtil->new();` `eval {` `$wfa_util->sendLog('INFO',` `"Connecting to the` `cluster: $DestinationCluster");` `}`
Registra ogni azione che richiede l'interazione con qualsiasi elemento esterno al comando, come Data ONTAP, VMware e WFA. Tutti i messaggi di log creati utilizzando la routine WFAUtil sendLog vengono memorizzati nel database WFA. Questi messaggi di log sono disponibili per il flusso di lavoro e il comando eseguiti.	Nessuno
Registrare tutti gli argomenti pertinenti passati alla routine chiamata.	Nessuno
Utilizzare i livelli di log appropriati.-Info, -Error, -warn e -Debug sono i vari livelli di log disponibili.	Nessuno

Linee guida	Esempio
Quando si accede al livello -Info, essere precisi e concisi. Non specificare dettagli di implementazione come il nome della classe e il nome della funzione nei messaggi di log. Descrivere la procedura esatta o l'errore esatto in inglese.	Il seguente frammento di codice mostra un esempio di messaggio valido e non valido: `$wfa_util->sendLog('WARN', "Removing volume: '.$VolumeName); # Good Message` `$wfa_util->sendLog('WARN', 'Invoking volume-destroy ZAPI: '.$VolumeName); # Bad message`

Linee guida per la gestione degli errori

È necessario conoscere le linee guida per la gestione degli errori durante la scrittura di uno script PowerShell o Perl per OnCommand Workflow Automation (Wfa).

Gestione degli errori PowerShell

Linee guida	Esempio
I parametri comuni aggiunti ai cmdlet dal runtime di PowerShell includono parametri di gestione degli errori come ErrorAction e WarningAction: • Il parametro ErrorAction determina il modo in cui un cmdlet deve reagire a un errore non terminante del comando. • Il parametro WarningAction determina il modo in cui un cmdlet deve reagire a un avviso proveniente dal comando. • Stop, SilentlyContinuare, Inquire e Continue sono i valori validi per i parametri ErrorAction e WarningAction. Per ulteriori informazioni, è possibile utilizzare `Get-Help about_CommonParameters` In PowerShell CLI.	ErrorAction: L'esempio seguente mostra come gestire un errore senza terminazione come errore finale: `New-NcIgroup-Name $IgroupName-Protocol $Protocol-Type$OSType-ErrorActionstop` WarningAction `New-VM-Name $VMName-VM $SourceVM-DataStore$DataStoreName-VMHost$VMHost-WarningActionSilentlyContinue`

Linee guida	Esempio
Utilizzare l'istruzione generale "try/catch" se il tipo di eccezione in entrata non è noto.	`try` `{` `"In Try/catch block"` `}` `catch` `{` `"Got exception"` `}`
Se si conosce il tipo di eccezione in entrata, utilizzare l'istruzione "Try/catch" specifica.	`try` `{` `"In Try/catch block"` `}` `catch[System.Net.WebExceptional],` `[System.IO.` `IOException]` `{` `"Got exception"` `}`
Utilizzare l'istruzione "finally" per rilasciare le risorse.	`try` `{` `"In Try/catch block"` `}` `catch` `{` `"Got exception"` `}` `finally` `{` `"Release resources"` `}`

Linee guida	Esempio
Utilizzare le variabili automatiche di PowerShell per accedere alle informazioni sulle eccezioni.	`try` `{` `Get-WFALogger -Info -message` `$("Creating` `Ipspace: " + $Ipspace)` `New-NaNetIpspace-Name $Ipspace` `}` `catch` `{` `Throw "Failed to create Ipspace.` `Message:` `" + $_.Exception.Message;` `}`

Gestione degli errori Perl

Linee guida	Esempio
Perl non include il supporto della lingua nativa per i blocchi try/catch. Utilizzare i blocchi di valutazione per il controllo e la gestione degli errori. Mantieni i blocchi di valutazione il più piccolo possibile.	``` 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", $@); ```

Convenzioni generali PowerShell e Perl per WFA

È necessario comprendere alcune convenzioni PowerShell e Perl utilizzate in WFA per creare script coerenti con quelli esistenti.

- Utilizzare variabili che consentono di chiarire le operazioni che si desidera eseguire nello script.
- Scrivere un codice leggibile che possa essere compreso senza commenti.
- Mantenere gli script e i comandi il più semplice possibile.
- Per gli script PowerShell:
  - Utilizzare i cmdlet quando possibile.
  - Richiamare il codice .NET quando non è disponibile alcun cmdlet.
- Per gli script Perl:

- Terminare sempre le istruzioni “die” con caratteri nuovi.

  In assenza di un carattere newline, viene stampato il numero della riga dello script, che non è utile per il debug dei comandi Perl eseguiti da WFA.

- Nel modulo “getopt”, rendere obbligatori gli argomenti stringa di un comando.

Moduli Perl in bundle con Windows

Alcuni moduli Perl sono forniti in bundle con la distribuzione Windows Active state Perl per OnCommand Workflow Automation (Wfa). È possibile utilizzare questi moduli Perl nel codice Perl per la scrittura dei comandi, solo se sono forniti con Windows.

La seguente tabella elenca i moduli di database Perl forniti in bundle con Windows per WFA.

Modulo di database	Descrizione
DBD::mysql	Driver di interfaccia per database Perl5 che consente di connettersi al database MySQL.
Prova::Tiny	Riduce al minimo gli errori comuni con i blocchi di valutazione.
XML::LibXML	Interfaccia a libxml2 che fornisce parser XML e HTML con interfacce DOM, SAX e XMLReader.
DBD::Alice	Driver di interfaccia database Perl5 per Cassandra che utilizza il linguaggio di query CQL3.

Considerazioni per l’aggiunta di moduli PowerShell e Perl personalizzati

Prima di aggiungere moduli PowerShell e Perl personalizzati a OnCommand Workflow Automation (Wfa), è necessario tenere presenti alcune considerazioni. I moduli personalizzati PowerShell e Perl consentono di utilizzare comandi personalizzati per la creazione di flussi di lavoro.

- Durante l’esecuzione dei comandi WFA, tutti i moduli PowerShell personalizzati vengono aggiunti alla directory di installazione di WFA `/Posh/modules` vengono importati automaticamente.
- Tutti i moduli Perl personalizzati aggiunti a `WFA/perl` Le directory sono incluse nella libreria @*Inc*.
- I moduli personalizzati PowerShell e Perl non vengono sottoposti a backup come parte dell’operazione di backup WFA.
- I moduli personalizzati PowerShell e Perl non vengono ripristinati come parte dell’operazione di ripristino WFA.

È necessario eseguire manualmente il backup dei moduli PowerShell e Perl personalizzati per poterli copiare in una nuova installazione WFA.

Il nome della cartella nella directory dei moduli deve essere uguale a quello del nome del modulo.

Cmdlet E funzioni DI WFA

OnCommand Workflow Automation (Wfa) offre diversi cmdlet PowerShell, oltre a funzioni PowerShell e Perl che è possibile utilizzare nei comandi Wfa.

È possibile visualizzare tutti i cmdlet e le funzioni PowerShell fornite dal server WFA utilizzando i seguenti comandi PowerShell:

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

È possibile visualizzare tutte le funzioni Perl fornite dal server WFA in `WFAUtil.pm` modulo. Le sezioni della guida, la guida ai cmdlet di WFA PowerShell e la guida ai metodi di WFA Perl, del modulo Guida di WFA Support Links consentono di accedere a tutti i cmdlet e alle funzioni di PowerShell e alle funzioni di Perl.

Moduli PowerShell e Perl WFA

È necessario conoscere i moduli PowerShell o Perl per OnCommand Workflow Automation (Wfa) per scrivere gli script per i flussi di lavoro.

Moduli PowerShell

Linee guida	Esempio
Utilizza il toolkit Data ONTAP PS per richiamare le API ogni volta che il toolkit è disponibile.	Il `Add VLAN` il comando utilizza il toolkit come segue: `Add-NaNetVlan-Interface $Interface-Vlans$VlanID`
Se non sono disponibili cmdlet nel toolkit Data ONTAP PS, utilizzare `Invoke-SSH` Per richiamare l'interfaccia CLI su Data ONTAP.	`Invoke-NaSsh-Name $ArrayName-Command "ifconfig -a"-Credential $Credentials`

Moduli Perl

Il modulo NaServer viene utilizzato nei comandi WFA. Il modulo NaServer consente di invocare le API Data ONTAP, utilizzate nella gestione attiva dei sistemi Data ONTAP.

Linee guida	Esempio
Utilizza il modulo NaServer per richiamare le API ogni volta che NetApp Manageability SDK è disponibile.	Nell'esempio seguente viene illustrato come utilizzare il modulo NaServer per riprendere un'operazione SnapMirror: `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',`

Linee guida	Esempio
Se non è disponibile un'API Data ONTAP, richiamare l'interfaccia utente di Data ONTAP utilizzando il metodo dell'utility executeSystemCli. ExecuteSystemCli non è supportato ed è attualmente disponibile solo per Data ONTAP in 7-Mode.	Nessuno

Considerazioni sulla conversione dei comandi PowerShell in Perl

Quando si convertono i comandi PowerShell in Perl, è necessario tenere presenti alcune considerazioni importanti, in quanto PowerShell e Perl hanno funzionalità diverse.

Tipi di input dei comandi

OnCommand Workflow Automation (WFA) consente ai progettisti di workflow di utilizzare array e hash come input per il comando quando si definisce un comando. Questi tipi di input non possono essere utilizzati quando il comando viene definito utilizzando Perl. Se si desidera che un comando Perl accetti gli input di array e hash, è possibile definire l'input come stringa nella finestra di progettazione. La definizione del comando può quindi analizzare l'input, che viene passato per creare una matrice o un hash secondo necessità. La descrizione dell'input descrive il formato in cui l'input è previsto.

```
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.
```

Dichiarazione PowerShell

I seguenti esempi mostrano come un input di array può essere passato a PowerShell e Perl. Gli esempi descrivono l'input di CronMonth, che specifica il mese in cui è pianificata l'esecuzione del lavoro cron. I valori validi sono numeri interi da -1 a 11. Il valore -1 indica che la pianificazione viene eseguita ogni mese. Qualsiasi altro valore indica un mese specifico, con 0 gennaio e 11 dicembre.

```
[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,
```

Dichiarazione Perl

```
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
}
```

Definizione del comando

Un'espressione a riga singola in PowerShell che utilizza un operatore di pipe potrebbe dover essere espansa in più blocchi di istruzioni in Perl per ottenere la stessa funzionalità. Un esempio di uno dei comandi di attesa è illustrato nella seguente tabella.

Dichiarazione PowerShell	Dichiarazione Perl
`# 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)` `}` `}`

Linee guida per gli elementi di base WFA

È necessario conoscere le linee guida per l'utilizzo degli elementi di base per l'automazione del flusso di lavoro.

Linee guida per SQL in WFA

È necessario conoscere le linee guida per l'utilizzo di SQL in OnCommand Workflow Automation (Wfa) per scrivere query SQL per Wfa.

SQL viene utilizzato nei seguenti punti di WFA:

- Query SQL per popolare gli input dell'utente per la selezione
- Query SQL per la creazione di filtri per filtrare gli oggetti di un tipo di voce dizionario specifico
- Dati statici nelle tabelle del database del parco giochi
- Tipo di origine dati personalizzata di tipo SQL in cui i dati devono essere estratti da un'origine dati esterna, ad esempio un database di gestione della configurazione personalizzato (CMDB).
- Query SQL per script di prenotazione e verifica

Linee guida	Esempio
Le parole chiave riservate di SQL devono essere in caratteri maiuscoli.	`SELECT` `  vserver.name` `FROM` `  cm_storage.vserver vserver`
I nomi delle tabelle e delle colonne devono essere costituiti da caratteri minuscoli.	Tabella: Aggregato Colonna: Used_space_mb
Separare le parole con un carattere di sottolineatura (_). Gli spazi non sono consentiti.	array_performance
Il nome della tabella è definito in singolare. Una tabella è una raccolta di una o più voci.	"function", non "functions"

Linee guida	Esempio
Utilizzare alias di tabella con nomi significativi nelle query SELEZIONATE.	``` 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 ```

Linee guida	Esempio
Se si deve fare riferimento a un parametro di input del filtro o a un parametro di input dell'utente in una query di filtro o utente, utilizzare la sintassi come '{inputVariableName}. È inoltre possibile utilizzare la sintassi per fare riferimento a un parametro di definizione del comando negli script di prenotazione e di verifica.	`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`
Utilizzare i commenti per query complesse. Alcuni degli stili di commento supportati nelle query sono i seguenti: • "--" fino alla fine della riga Uno spazio è obbligatorio dopo il secondo trattino in questo stile di commento. • Da un carattere "` n.`" fino alla fine della riga • Da "/*" to the following "`sequenza */`	`/*` `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`

Linee guida per le funzioni WFA

È possibile creare funzioni per incapsulare la logica comunemente utilizzata e più complessa in una funzione denominata, quindi riutilizzare la funzione come valori dei parametri di comando o valori dei parametri di filtro in OnCommand Workflow Automation (Wfa).

Linee guida	Esempio
Utilizzare il caso Camel per il nome di una funzione.	CalcolateVolumeSize
I nomi delle variabili devono essere in inglese normale e correlati alla funzionalità della funzione.	SplitByDelimiter
Non utilizzare abbreviazioni.	CalculateVolumeSize, *not* calcVolSize
Le funzioni vengono definite utilizzando MVFLEX Expression Language (MVEL).	Nessuno
La definizione della funzione deve essere specificata in base alle linee guida ufficiali del linguaggio di programmazione Java.	Nessuno

Linee guida per le voci del dizionario WFA

È necessario conoscere le linee guida per la creazione di voci dizionario in OnCommand Workflow Automation (WFA).

Linee guida	Esempio
I nomi delle voci del dizionario devono contenere solo caratteri alfanumerici e caratteri di sottolineatura.	Licenza_cluster Switch_23
I nomi delle voci del dizionario devono iniziare con un carattere maiuscolo. Inizia ogni parola del nome con un carattere maiuscolo e separa ogni parola con un carattere di sottolineatura (_).	Volume Array_License
I nomi degli attributi delle voci del dizionario non devono includere il nome della voce del dizionario.	Nessuno
Gli attributi e i riferimenti in una voce del dizionario devono essere in caratteri minuscoli.	aggregato, size_mb
Separare le parole con un carattere di sottolineatura. Gli spazi non sono consentiti.	pool_risorse

Linee guida	Esempio
Le voci del dizionario non possono includere riferimenti provenienti da uno schema diverso. Quando una voce del dizionario richiede un riferimento incrociato a un oggetto in uno schema diverso, assicurarsi che tutte le chiavi naturali dell'oggetto a cui si fa riferimento siano presenti nella voce del dizionario.	La voce del dizionario Array_Performance richiede tutte le chiavi naturali della voce del dizionario Array come attributi diretti.
Utilizzare i tipi di dati appropriati per gli attributi.	Nessuno
Utilizzare il tipo di dati Long per gli attributi relativi alla dimensione o allo spazio.	Size_mb e available_size_mb nella voce storage.Volume dictionary
Utilizzare Enum quando un attributo ha un set di valori fisso.	raid_type in storage.Volume dictionary entry
Impostare "da memorizzare nella cache" come true per un attributo o un riferimento quando un'origine dati fornisce un valore per tale attributo o riferimento.per l'origine dati Active IQ Unified Manager, aggiungere attributi memorizzabili nella cache se l'origine dati è in grado di fornirlo.	Nessuno
Impostare "CAN be null" come true se l'origine dati che fornisce il valore per questo attributo o riferimento può restituire NULL.	Nessuno
Fornire una descrizione significativa di ciascun attributo e riferimento.la descrizione viene visualizzata nei dettagli del comando durante la progettazione di un flusso di lavoro.	Nessuno
Non utilizzare "id" come nome di un attributo nelle voci del dizionario.è riservato all'utilizzo interno di WFA.	Nessuno

Informazioni correlate

Riferimenti al materiale di apprendimento

Linee guida per i comandi

È necessario conoscere le linee guida per la creazione dei comandi in OnCommand Workflow Automation (Wfa).

Linee guida	Esempio
Utilizzare un nome facilmente identificabile per i comandi.	`Create Qtree`
Utilizzare gli spazi per delimitare le parole e ciascuna parola deve iniziare con un carattere maiuscolo.	`Create Volume`
Fornire una descrizione per spiegare la funzionalità del comando, compreso il risultato previsto dei parametri opzionali.	Nessuno
Per impostazione predefinita, il timeout per i comandi standard è di 600 secondi. Il timeout predefinito viene impostato durante la creazione del comando. Modificare il valore predefinito solo se il completamento del comando potrebbe richiedere più tempo.	`Create Volume` comando
In caso di operazioni a esecuzione prolungata, creare due comandi, uno per richiamare l'operazione a esecuzione prolungata e l'altro per segnalare periodicamente l'avanzamento dell'operazione. Il primo comando deve essere un `Standard Execution` il tipo di comando e il secondo dovrebbe essere `Wait for Condition` tipo di comando.	`Create VSM` e. `Wait for VSM` comandi
Inserire il prefisso `Wait for condition` Nomi dei comandi con "wait" per una facile identificazione.	`Wait for CM Volume Move`
Utilizzare un intervallo di attesa appropriato per i comandi "wait for condition". Il valore specificato regola l'intervallo di esecuzione del comando di polling per verificare se l'operazione a esecuzione prolungata è completa.	intervallo di campionamento di 60 secondi per `Wait for VSM` comando
Per `Wait for condition` utilizzare un timeout appropriato in base al tempo previsto per il completamento dell'operazione a lungo termine. Il tempo previsto potrebbe essere notevolmente più lungo se l'operazione comporta il trasferimento dei dati su una rete.	Il completamento di un trasferimento di riferimento VSM può richiedere molti giorni. Pertanto, il timeout specificato è di 6 giorni.

Rappresentazione di stringhe

La rappresentazione stringa di un comando visualizza i dettagli di un comando in una progettazione del flusso di lavoro durante la pianificazione e l'esecuzione. Nella rappresentazione stringa di un comando possono essere utilizzati solo i parametri del comando.

Linee guida	Esempio
Evitare di utilizzare attributi che non hanno alcun valore. Un attributo senza valore viene visualizzato come NA.	VolName 10.68.66.212[NA]aggr1/testVol7
Separare le diverse voci nella rappresentazione della stringa utilizzando i seguenti delimitatori: [ ] , / :	*ArrayName[ArrayIp]*
Fornire etichette significative per ogni valore nella rappresentazione della stringa.	*Volume name=VolumeName*

Linguaggio di definizione dei comandi

I comandi possono essere scritti utilizzando i seguenti linguaggi di scripting supportati:

- PowerShell
- Perl

Definizione del parametro del comando

I parametri dei comandi sono descritti in base a Nome, Descrizione, tipo, un valore predefinito per il parametro e se il parametro è obbligatorio. Il tipo di parametro può essere String, Boolean, Integer, Long, Double, Enum, DateTime, Capacity, Array, Hashtable, Password o XmlDocument. Sebbene i valori per la maggior parte dei tipi siano intuitivi, i valori per Array e Hashtable devono essere in un formato particolare, come descritto nella tabella seguente:

Linee guida	Esempio
Assicurarsi che il valore di un tipo di input Array sia un elenco di valori, separati da una virgola.	`[parameter(Mandatory=$false, HelpMessage="Months in which the schedule executes.")] [array]$CronMonths` L'input viene passato come segue: 0,3,6,9
Assicurarsi che il valore di un tipo di input Hashtable sia un elenco di coppie chiave=valore, separate da punto e virgola.	`[parameter(Mandatory=$false, HelpMessage="Volume names and size (in MB)")] [hashtable]$VolumeNamesAndSize` L'input viene passato come segue: Volume1=100;Volume2=250;Volume3=50

Linee guida per i flussi di lavoro

È necessario conoscere le linee guida per la creazione o la modifica di un flusso di lavoro predefinito per OnCommand Workflow Automation (WFA).

Linee guida generali

Linee guida	Esempio
Assegnare un nome al flusso di lavoro in modo che rifletta l'operazione eseguita dall'operatore di storage.	`Create a CIFS Share`
Per i nomi dei flussi di lavoro, scrivere in maiuscolo la lettera iniziale della prima parola e ogni parola che è un oggetto. Lettere maiuscole per abbreviazioni e acronimi.	Volume Qtree Creare una condivisione CIFS Qtree Data ONTAP in cluster
Per le descrizioni del flusso di lavoro, includere tutte le fasi importanti del flusso di lavoro, inclusi eventuali prerequisiti, risultato del flusso di lavoro o aspetti condizionali dell'esecuzione.	Vedere la descrizione del flusso di lavoro di esempio `Create VMware NFS Datastore on Clustered Data ONTAP Storage`, che include i prerequisiti.
Impostare "Ready for Production" su `true` solo quando il flusso di lavoro è pronto per la produzione e può essere visualizzato nella pagina del portale.	Nessuno
Per impostazione predefinita, impostare "considerare elementi riservati" su true. Durante l'anteprima di un flusso di lavoro per l'esecuzione, il pianificatore WFA prende in considerazione tutti gli oggetti riservati insieme agli oggetti esistenti nel database della cache. Gli effetti di altri flussi di lavoro pianificati o di flussi di lavoro in esecuzione in parallelo vengono presi in considerazione quando si pianifica un flusso di lavoro specifico, se questa opzione è impostata su `true`.	• Scenario 1 Workflow 1 crea un volume e viene pianificato per l'esecuzione una settimana dopo. Workflow 2 crea qtree o LUN nei volumi ricercati e, se il workflow 2 viene eseguito entro un giorno, è necessario disattivare "considerare elementi riservati" per il workflow 2 per evitare che prenda in considerazione il volume che deve essere creato in una settimana. • Scenario 2 Il flusso di lavoro 1 utilizza `Create Volume` comando. Se esiste un flusso di lavoro pianificato 2 che consuma 100 GB da un aggregato, il flusso di lavoro 1 deve considerare i requisiti per il flusso di lavoro 2 durante la pianificazione.

Linee guida	Esempio
Per impostazione predefinita, “Enable Element existence validation” è impostato su `true`.	• Scenario 1 Se si crea un flusso di lavoro che prima rimuove un volume in base al nome utilizzando il comando `Remove Volume` solo se il volume esiste e lo crea nuovamente utilizzando un altro comando, ad esempio `Create Volume` oppure `Clone Volume`, il flusso di lavoro non deve utilizzare questo flag. L’effetto della rimozione del volume non sarà disponibile per `Create volume` di conseguenza, il flusso di lavoro non riesce. • Scenario 2 Il `Create Volume` il comando viene utilizzato in un flusso di lavoro con un nome specifico come “vol198”. Se questa opzione è impostata su true, WFA Planner verifica durante la pianificazione se un volume con tale nome esiste nell’array specificato. Se il volume esiste, il flusso di lavoro non riesce durante la pianificazione.
Quando lo stesso comando viene selezionato più di una volta in un flusso di lavoro, fornire i nomi di visualizzazione appropriati per le istanze di comando.	Il flusso di lavoro di esempio “creazione, mappatura e protezione LUN con SnapVault” utilizza `Create Volume` comando due volte. Tuttavia, utilizza i nomi visualizzati come `Create Primary Volume` e. `Create Secondary Volume` appropriato per il volume primario e il volume di destinazione mirrorato.

Input dell’utente

Linee guida	Esempio
Nomi: • Iniziare il nome con il carattere "``". • Utilizzare una lettera maiuscola all’inizio di ciascuna parola. • Utilizzare lettere maiuscole per tutti i termini e le abbreviazioni. • Non utilizzare i caratteri di sottolineatura.	`$Array` `$VolumeName`

Linee guida	Esempio
Nomi visualizzati: • Utilizzare una lettera maiuscola all'inizio di ciascuna parola. • Separare le parole con spazi. • Se gli input hanno unità specifiche, specificare direttamente l'unità tra parentesi nel nome visualizzato.	`Volume Name` `Volume Size (MB)`
Descrizioni: • Fornire una descrizione significativa per ogni input dell'utente. • Fornire esempi quando necessario. Questa operazione deve essere eseguita in particolare quando si prevede che l'input dell'utente sia in un formato specifico. Le descrizioni degli input dell'utente vengono visualizzate come descrizioni dei comandi per gli input dell'utente durante l'esecuzione del flusso di lavoro.	Iniziatori da aggiungere a un "iGroup". Ad esempio, IQN o WWPN dell'iniziatore.
Type (tipo): Selezionare Enum (tipo) se si desidera limitare l'input a un set di valori specifico.	Protocollo: "iscsi", "fcp", "mixed"
Type (tipo): Selezionare Query come tipo quando l'utente può selezionare i valori disponibili nella cache WFA.	Array: Tipo DI QUERY con query come segue: `SELECT` `ip, name` `FROM` `storage.array`
Type (tipo): Contrassegna l'input dell'utente come bloccato quando l'input dell'utente deve essere limitato ai valori ottenuti da una query o deve essere limitato solo ai tipi di enum supportati.	Array: Tipo di query bloccata: È possibile selezionare solo gli array nella cache. Protocollo: Tipo di enum bloccato con valori validi come iscsi, fcp, misto. Non sono supportati valori diversi da quelli validi.
Type: Query TypeAggiungi colonne aggiuntive come valori di ritorno nella query quando aiuta l'operatore di storage a scegliere l'input dell'utente corretto.	Aggregato di dollari: Fornire nome, dimensione totale e dimensione disponibile in modo che l'operatore conosca gli attributi prima di selezionare l'aggregato.

Linee guida	Esempio
Type: Query TypeQuery SQL per gli input dell'utente può fare riferimento a qualsiasi altro input dell'utente che lo precede. Questo può essere utilizzato per limitare i risultati di una query basata su altri input dell'utente, come unità vFiler di un array, volumi di un aggregato, LUN in una macchina virtuale di storage (SVM).	Nel flusso di lavoro di esempio `Create a Clustered Data ONTAP Volume`, La query per VserverName è la seguente: `SELECT` `vserver.name` `FROM` `cm_storage.cluster cluster,` `cm_storage.vserver vserver` `WHERE` `vserver.cluster_id = cluster.id` `AND cluster.name = '${ClusterName}'` `AND vserver.type = 'cluster'` `ORDER BY` `vserver.name ASC` La query si riferisce a{ClusterName}, dove NomeClusterNomeClusterNomeClusterNomeCluster NomeClusterNomeClusterNomeClusterNomeCluster NomeClusterNomeClusterNomeClusterNomeCluster NomeClusterNome
Type (tipo): Utilizzare il tipo booleano con valori come "true, false" per gli input utente di natura booleano. Ciò consente di scrivere espressioni interne nella progettazione del flusso di lavoro utilizzando direttamente l'input dell'utente. Ad esempio, NomeUtente anziché NomeUtente == '`Sì'.	`$CreateCIFSShare`: Tipo booleano con valori validi come "true" o "false"
Type:per il tipo di stringa e numero, utilizzare espressioni regolari nella colonna valori quando si desidera convalidare il valore con formati specifici. Utilizzare espressioni regolari per gli input di indirizzo IP e maschera di rete.	L'input utente specifico della posizione può essere espresso come "[A-Z][A-Z]`-0[1-9]". Questo input utente accetta valori come "`US-01", "NB-02", ma non "nb-00".
Type (tipo): Per il tipo di numero, è possibile specificare una convalida basata sull'intervallo nella colonna Values (valori).	Per il numero di LUN da creare, la voce nella colonna valori è 1-20.
Group (Gruppo): Raggruppa gli input utente correlati nei bucket appropriati e assegna un nome al gruppo.	"storage Details" per tutti gli input utente relativi allo storage. "dAtastore Details" per tutti gli input utente relativi a VMware.

Linee guida	Esempio
Obbligatorio: Se il valore di qualsiasi input dell'utente è necessario per l'esecuzione del flusso di lavoro, contrassegnare l'input dell'utente come obbligatorio. In questo modo, la schermata di input dell'utente accetta in modo manuale l'input dell'utente.	"` VolumeName`" nel flusso di lavoro "Create NFS Volume".
Default value (valore predefinito): Se un input utente ha un valore predefinito che può funzionare per la maggior parte delle esecuzioni del flusso di lavoro, fornire i valori. Ciò consente all'utente di fornire un minor numero di input durante l'esecuzione, se l'impostazione predefinita serve a tale scopo.	Nessuno

Costanti, variabili e parametri di ritorno

Linee guida	Esempio
Costanti: Consente di definire le costanti quando si utilizza un valore comune per la definizione dei parametri a più comandi.	*AGGREGATE_OVERCOMMITMENT_THRESHOLD* in `Create, map, and protect LUNs with SnapVault sample workflow.`
Costanti:nomi • Utilizzare una lettera maiuscola all'inizio di ciascuna parola. • Utilizzare lettere maiuscole per tutti i termini e le abbreviazioni. • Non utilizzare i caratteri di sottolineatura. • Utilizzare lettere maiuscole per tutte le lettere di nomi costanti.	*AGGREGATE_USED_SPACE_THRESHOLD* *ActualVolumeSizeInMB*
Variables (variabili): Fornire un nome a un oggetto definito in una delle caselle dei parametri di comando. Le variabili sono nomi generati automaticamente e possono essere modificate.	Nessuno
Variabili: I nomi utilizzano caratteri minuscoli per i nomi delle variabili.	volume1 cifs_share
Parametri di ritorno: Utilizzare i parametri di ritorno quando la pianificazione e l'esecuzione del workflow devono restituire alcuni valori calcolati o selezionati durante la pianificazione. I valori vengono resi disponibili nella modalità di anteprima quando il flusso di lavoro viene eseguito anche da un servizio Web.	Aggregato: Se l'aggregato viene selezionato utilizzando la logica di selezione delle risorse, l'aggregato effettivo selezionato può essere definito come parametro di ritorno.

Linee guida per la creazione di script di convalida per i tipi di sistema remoto

È necessario conoscere le linee guida per la creazione di script di convalida utilizzati per testare i tipi di sistema remoto definiti in OnCommand Workflow Automation (Wfa).

- Lo script Perl creato deve essere simile allo script di esempio fornito nella finestra Validation script (script di convalida).
- L'output dello script di convalida deve essere simile a quello dello script di esempio.

Script di convalida di esempio

```
# 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;
}
```

Linee guida per la creazione di tipi di origine dati

È necessario conoscere le linee guida per la creazione di tipi di origine dati utilizzati per definire origini dati personalizzate per OnCommand Workflow Automation (Wfa).

È possibile definire un tipo di origine dati utilizzando uno dei seguenti metodi:

- SQL: È possibile utilizzare le linee guida SQL di Wfa per definire query selezionate da origini dati basate su un database esterno.
- SCRIPT: È possibile scrivere uno script PowerShell che fornisce i dati per uno schema specifico di voci del dizionario.

Le linee guida per la creazione dei tipi di origine dati sono le seguenti:

- Per creare lo script, è necessario utilizzare il linguaggio PowerShell.
- Lo script PowerShell deve fornire l'output per ogni voce del dizionario nella directory di lavoro corrente.
- I file di dati devono essere denominati `dictionary_entry.csv`, dove il nome della voce del dizionario deve essere in caratteri minuscoli.

  Il tipo di origine dati predefinito che raccoglie informazioni da Performance Advisor utilizza un tipo di origine dati basato SU SCRIPT. I file di output sono denominati `array_performance.csv` e. `aggregate_performance.csv`.

- Il `.csv` il file deve includere il contenuto nell'ordine esatto degli attributi delle voci del dizionario.

  Una voce del dizionario include gli attributi nel seguente ordine: Array_ip, data, giorno, ora, cpu_busy,

total_ops_per_sec, disk_throughput_per_sec.

Lo script PowerShell aggiunge i dati a `.csv` file nello stesso ordine.

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

- È necessario utilizzare la codifica per garantire che l'output dei dati dallo script sia caricato correttamente nella cache WFA.
- Utilizzare il pulsante N. durante l'immissione di un valore nullo in `.csv` file.

Informazioni sul copyright

Informazioni sul marchio commerciale