Diretrizes de codificação para WFA

OnCommand Workflow Automation 5.1

NetApp
October 22, 2024

Índice

Diretrizes de codificação para WFA

Você deve entender as diretrizes gerais de codificação do OnCommand Workflow Automation (WFA), convenções de nomenclatura e recomendações sobre a criação de vários blocos de construção, como filtros, funções, comandos e fluxos de trabalho.

Diretrizes para variáveis

Você deve estar ciente das diretrizes para variáveis PowerShell e Perl no OnCommand Workflow Automation (WFA) ao criar um comando ou um tipo de fonte de dados.

Variáveis do PowerShell

Diretrizes	Exemplo
Para parâmetros de entrada de script: • Use Pascal Case. • Não utilize sublinhados. • Não utilize abreviaturas.	`$VolumeName` `$AutoDeleteOptions` `$Size`
Para variáveis internas de script: • Use Camel Case. • Não utilize sublinhados. • Não utilize abreviaturas.	`$newVolume` `$qtreeName` `$time`
Para funções: • Use Pascal Case. • Não utilize sublinhados. • Não utilize abreviaturas.	`GetVolumeSize`
Nomes de variáveis não são sensíveis a maiúsculas e minúsculas. No entanto, para melhorar a legibilidade, você não deve usar letras maiúsculas diferentes para o mesmo nome.	`$variable` é o mesmo que `$Variable.`
Os nomes das variáveis devem estar em inglês simples e devem estar relacionados à funcionalidade do script.	Use `$name` e não `$a.`
Declare o tipo de dados para cada variável, explicitamente.	[string]nome [int]tamanho

Diretrizes	Exemplo
Não use carateres especiais (! a e % , .) e espaços.	Nenhum
Não use palavras-chave reservadas do PowerShell.	Nenhum
Agrupe os parâmetros de entrada colocando os parâmetros obrigatórios primeiro seguidos pelos parâmetros opcionais.	`param(` `[parameter(Mandatory=$true)]` `[string]$Type,` `[parameter(Mandatory=$true)]` `[string]$Ip,` `[parameter(Mandatory=$false)]` `[string]$VolumeName` `)`
Comente todas as variáveis de entrada usando *`HelpMessage`* anotação com uma mensagem de ajuda significativa.	`[parameter(Mandatory=$false,HelpMessage="LUN to map")]` `[string]$LUNName`
Não use "'Filer'" como um nome de variável; use "'Array'" em vez disso.	Nenhum
`` `_ValidateSet_`Use a anotação nos casos em que o argumento obtém valores enumerados. Isso se traduz automaticamente para o tipo de dados Enum para o parâmetro. ``	`[parameter(Mandatory=$false,HelpMessage="Volume state")]` `[ValidateSet("online","offline","restricted")]` `[string]$State`
Adicione um alias a um parâmetro que termine com "'_capacity" para indicar que o parâmetro é do tipo capacidade.	O comando "Create volume" usa aliases da seguinte forma: `[parameter(Mandatory=$false,HelpMessage="Volume increment size in MB")]` `[Alias("AutosizeIncrementSize_Capacity")]` `[int]$AutosizeIncrementSize`

Diretrizes	Exemplo
Adicione um alias a um parâmetro que termine com "'_Password" para indicar que o parâmetro é do tipo de senha.	``` 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) ```

Variáveis Perl

Diretrizes	Exemplo
Para parâmetros de entrada de script: • Use Pascal Case. • Não utilize sublinhados. • Não utilize abreviaturas.	`$VolumeName` `$AutoDeleteOptions` `$Size`
Não use abreviações para variáveis internas de script.	`$new_volume` `$qtree_name` `$time`
Não utilize abreviaturas para funções.	`get_volume_size`
Nomes de variáveis são sensíveis a maiúsculas e minúsculas. Para melhorar a legibilidade, você não deve usar letras maiúsculas diferentes para o mesmo nome.	`$variable` não é o mesmo que `$Variable`.
Os nomes das variáveis devem estar em inglês simples e devem estar relacionados à funcionalidade do script.	Use `$name` e não `$a`.
Agrupe os parâmetros de entrada colocando os parâmetros obrigatórios primeiro, seguidos pelos parâmetros opcionais.	Nenhum

Diretrizes	Exemplo
Na função GetOptions, declare explicitamente o tipo de dados de cada variável para parâmetros de entrada.	``` GetOptions("Name=s"=>\$Name, "Size=i"=>\$Size) ```
Não use "'Filer'" como um nome de variável; use "'Array'" em vez disso.	Nenhum
Perl não inclui a *`ValidateSet`* anotação para valores enumerados. Use declarações explícitas "if" para casos em que argumento obtém valores enumerados.	``` if (defined$SpaceGuarantee&&!($SpaceG uaranteeeq'none' ```
	$SpaceGuaranteeeq’volume'
	$SpaceGuaranteeeq’file')) { die’Illegal SpaceGuarantee argument: \".$SpaceGuarantee.'\"; } ----
Todos os comandos Perl WFA devem usar o pragma "strict" para desencorajar o uso de construções inseguras para variáveis, referências e sub-rotinas.	``` use strict; # the above is equivalent to use strictvars; use strictsubs; use strictrefs; ```
Todos os comandos Perl WFA devem usar os seguintes módulos Perl: • Getopt Isso é usado para especificar parâmetros de entrada. • WFAUtil Isso é usado para funções de utilitário que são fornecidas para Registro de comandos, relatório do progresso do comando, conexão a controladores de array e assim por diante.	``` use Getopt::Long; use NaServer; use WFAUtil; ```

Diretrizes para indentação

Você deve estar ciente das diretrizes para indentação ao escrever um script PowerShell

ou Perl para OnCommand Workflow Automation (WFA).

Diretrizes	Exemplo
Um separador é igual a quatro espaços vazios.	
Use abas e chaves para mostrar o início e o fim de um bloco.	Script do PowerShell `if ($pair.length-ne 2) { throw "Got wrong input data" }` Script Perl `if (defined $MaxDirectorySize) { # convert from MBytes to Bytes my $MaxDirectorySizeBytes = $MaxDirectorySize * 1024 * 1024; }`
Adicione linhas em branco entre conjuntos de operações ou blocos de código.	`$options=$option.trim(); $pair=$option.split(" "); Get-WFAlogger -Info -messages $("split options: "+ $Pair)`

Diretrizes para comentários

Você deve estar ciente das diretrizes para comentários do PowerShell e Perl em seus scripts para OnCommand Workflow Automation (WFA).

Comentários do PowerShell

Diretrizes	Exemplo
Use o caractere nº para um comentário de linha única.	`# Single line comment` `$options=$option.trim();`
Use o caractere nº para um comentário de fim de linha.	`$options=$option.trim(); # End of line` `comment`
Use os carateres no e no> para um comentário em bloco.	`<#` `This is` `a` `block comment` `#>` `$options=$option.trim();`

Perl comentários

Diretrizes	Exemplo
Use o caractere nº para um comentário de linha única.	`# convert from MBytes to Bytes` `my $MaxDirectorySizeBytes =` `$MaxDirectorySize *` `1024 * 1024;`
Use o caractere nº para o comentário de fim de linha.	`my $MaxDirectorySizeBytes =` `$MaxDirect` `orySiZe * 1024 * 1024; # convert` `to Bytes`

Diretrizes	Exemplo
Use o caractere nº em cada linha com um número vazio no início e no fim para criar uma borda de comentário para comentários multilinhas.	```# # 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 #```
Não inclua código comentado e morto nos comandos DO WFA. No entanto, para fins de teste, você pode usar o mecanismo DE Documentação Velha simples (POD) para comentar o código.	```=begin comment # Set deduplication if(defined $Deduplication && $Deduplication eq "enabled") { $wfaUtil->sendLog("Enabling Deduplication"); } =end comment =cut```

Diretrizes para o Registro de logs

Você deve estar ciente das diretrizes para o Registro ao escrever um script PowerShell ou Perl para OnCommand Workflow Automation (WFA).

Log do PowerShell

Diretrizes	Exemplo
Use o cmdlet Get-WFALogger para Registro.	```Get-WFALogger -Info -message “Creating volume”```

Diretrizes	Exemplo
Registre todas as ações que requerem interação com pacotes internos, como Data ONTAP, VMware e PowerCLI. Todas as mensagens de log estão disponíveis em Logs de execução no histórico de status de execução dos fluxos de trabalho.	Nenhum
Registre cada argumento relevante que é passado para pacotes internos.	Nenhum
Use níveis de log apropriados ao usar o cmdlet Get-WFALogger, dependendo do contexto de uso. -Info, -Error, -WARN e -Debug são os vários níveis de log disponíveis. Se um nível de log não for especificado, então o nível de log padrão é Debug.	Nenhum

Registo Perl

Diretrizes	Exemplo
Use o sendLog do WFAUtil para Registrar.	`my wfa_util = WFAUtil->new();` `eval {` `$wfa_util->sendLog('INFO',` `"Connecting to the` `cluster: $DestinationCluster");` `}`
Registre todas as ações que exigem interação com qualquer coisa externa ao comando, como Data ONTAP, VMware e WFA. Todas as mensagens de log criadas usando a rotina sendLog do WFAUtil são armazenadas no banco de dados DO WFA. Essas mensagens de log estão disponíveis para o fluxo de trabalho e comando executados.	Nenhum
Registre cada argumento relevante passado para a rotina que foi chamada.	Nenhum
Use níveis de log apropriados.-Info, -Error, -WARN e -Debug são os vários níveis de log disponíveis.	Nenhum

Diretrizes	Exemplo
Ao Registrar no nível -Info, seja preciso e conciso. Não especifique detalhes de implementação, como nome da classe e nome da função em mensagens de log. Descreva a etapa exata ou o erro exato em inglês simples.	O snippet de código a seguir mostra um exemplo de uma mensagem boa e uma mensagem ruim: `$wfa_util->sendLog('WARN', "Removing volume: '.$VolumeName); # Good Message` `$wfa_util->sendLog('WARN', 'Invoking volume-destroy ZAPI: '.$VolumeName); # Bad message`

Diretrizes para tratamento de erros

Você deve estar ciente das diretrizes para manipulação de erros ao escrever um script PowerShell ou Perl para OnCommand Workflow Automation (WFA).

Manipulação de erros do PowerShell

Diretrizes	Exemplo
Parâmetros comuns adicionados aos cmdlets pelo PowerShell runtime incluem parâmetros de manipulação de erros, como ErrorAction e WarningAction: • O parâmetro ErrorAction determina como um cmdlet deve reagir a um erro que não encerra o comando. • O parâmetro WarningAction determina como um cmdlet deve reagir a um aviso do comando. • Stop, SilentlyContinuar, inquirir e continuar são os valores válidos para os parâmetros ErrorAction e WarningAction. Para obter mais informações, você pode usar o `Get-Help about_CommonParameters` comando na CLI do PowerShell.	ErrorAction: O exemplo a seguir mostra como lidar com um erro não-terminação como um erro de terminação: `New-NcIgroup-Name $IgroupName-Protocol $Protocol-Type$OSType-ErrorActionstop` WarningAction `New-VM-Name $VMName-VM $SourceVM-DataStore$DataStoreName-VMHost$VMHost-WarningActionSilentlyContinue`

Diretrizes	Exemplo
Use a instrução geral "try/catch" se o tipo da exceção recebida for desconhecido.	`try` `{` `"In Try/catch block"` `}` `catch` `{` `"Got exception"` `}`
Use a instrução específica "try/catch" se o tipo da exceção recebida for conhecido.	`try` `{` `"In Try/catch block"` `}` `catch[System.Net.WebExceptional],` `[System.IO.` `IOException]` `{` `"Got exception"` `}`
Use a declaração "finalmente" para liberar recursos.	`try` `{` `"In Try/catch block"` `}` `catch` `{` `"Got exception"` `}` `finally` `{` `"Release resources"` `}`

Diretrizes	Exemplo
Use variáveis automáticas do PowerShell para acessar informações sobre exceções.	``` try { Get-WFALogger -Info -message $("Creating Ipspace: " + $Ipspace) New-NaNetIpspace-Name $Ipspace } catch { Throw "Failed to create Ipspace. Message: " + $_.Exception.Message; } ```

Manipulação de erros Perl

Diretrizes	Exemplo
Perl não inclui suporte de linguagem nativa para blocos try/catch. Use blocos eval para verificar e lidar com erros. Mantenha os blocos eval o mais pequenos possível.	``` 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", $@); ```

Convenções gerais do PowerShell e Perl para O WFA

Você deve entender certas convenções PowerShell e Perl que são usadas no WFA para criar scripts que são consistentes com scripts existentes.

- Use variáveis que ajudam a esclarecer o que você quer que o script faça.
- Escreva código legível que pode ser entendido sem comentários.
- Mantenha os scripts e comandos o mais simples possível.
- Para scripts do PowerShell:
  - Use cmdlets sempre que possível.
  - Invoque o código .NET quando não houver cmdlet disponível.
- Para scripts Perl:

- Sempre termine as declarações "die" com carateres de nova linha.

  Na ausência de um caractere de nova linha, o número da linha do script é impresso, o que não é útil para depurar comandos Perl executados pelo WFA.

- No módulo "getopt", torne obrigatório os argumentos de string para um comando.

Módulos Perl empacotados com Windows

Alguns módulos Perl são empacotados com a distribuição Perl de estado ativo do Windows para OnCommand Workflow Automation (WFA). Você pode usar esses módulos Perl em seu código Perl para escrever comandos, apenas se eles estiverem empacotados com o Windows.

A tabela a seguir lista os módulos de banco de dados Perl que são empacotados com o Windows para WFA.

Módulo da base de dados	Descrição
DBD::mysql	Driver de interface de banco de dados Perl5 que permite que você se conete ao banco de dados MySQL.
Tente::Tiny	Minimiza erros comuns com blocos de avaliação.
XML::libxml	Interface para libxml2 que fornece analisadores XML e HTML com interfaces Dom, SAX e XMLReader.
DBD::Cassandra	Driver de interface de banco de dados Perl5 para Cassandra que usa a linguagem de consulta CQL3.

Considerações para adicionar módulos personalizados PowerShell e Perl

Você deve estar ciente de certas considerações antes de adicionar módulos personalizados PowerShell e Perl ao OnCommand Workflow Automation (WFA). Os módulos personalizados PowerShell e Perl permitem que você use comandos personalizados para criar fluxos de trabalho.

- Durante a execução dos comandos WFA, todos os módulos personalizados do PowerShell são adicionados ao diretório de instalação DO WFA `/Posh/modules` são importados automaticamente.
- Todos os módulos Perl personalizados adicionados ao `WFA/perl` diretório estão incluídos na biblioteca _ Inc_.
- Os módulos personalizados PowerShell e Perl não são copiados como parte da operação de backup DO WFA.
- Os módulos personalizados PowerShell e Perl não são restaurados como parte da operação de restauração DO WFA.

Você deve fazer o backup manual de módulos personalizados do PowerShell e Perl para copiá-los para uma

nova instalação DO WFA.

O nome da pasta no diretório dos módulos deve ser o mesmo do nome do módulo.

Cmdlets e funções DO WFA

O OnCommand Workflow Automation (WFA) fornece vários cmdlets do PowerShell, bem como funções do PowerShell e Perl que você pode usar em seus comandos DO WFA.

Você pode exibir todos os cmdlets e funções do PowerShell fornecidos pelo SERVIDOR WFA usando os seguintes comandos do PowerShell:

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

Você pode visualizar todas as funções Perl fornecidas pelo servidor WFA `WFAUtil.pm` no módulo. As seções de ajuda, os cmdlets DO WFA PowerShell e a ajuda dos métodos Perl do WFA, dos links de suporte DO módulo Ajuda DO WFA permitem o acesso a todos os cmdlets e funções do PowerShell e às funções Perl.

Módulos do PowerShell e Perl WFA

Você deve estar ciente dos módulos PowerShell ou Perl para OnCommand Workflow Automation (WFA) para escrever scripts para seus fluxos de trabalho.

Módulos do PowerShell

Diretrizes	Exemplo
Use o Kit de Ferramentas PS do Data ONTAP para invocar APIs sempre que o kit de ferramentas estiver disponível.	O `Add VLAN` comando usa o kit de ferramentas da seguinte forma: `Add-NaNetVlan-Interface $Interface-Vlans$VlanID`
Se não houver cmdlets disponíveis no Kit de Ferramentas PS do Data ONTAP, use o `Invoke-SSH` comando para chamar a CLI no Data ONTAP.	`Invoke-NaSsh-Name $ArrayName-Command "ifconfig -a"-Credential $Credentials`

Módulos Perl

O módulo NaServer é usado nos comandos WFA. O módulo NaServer permite a invocação de APIs Data ONTAP, que são usadas no gerenciamento ativo de sistemas Data ONTAP.

Diretrizes	Exemplo
Use o módulo NaServer para invocar APIs sempre que o SDK de gerenciamento do NetApp estiver disponível.	O exemplo a seguir mostra como o módulo NaServer é usado para uma operação RESUME 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', "Result of resume: $result");`

Diretrizes	Exemplo
Se uma API do Data ONTAP não estiver disponível, chame a CLI do Data ONTAP usando o método do utilitário executeSystemCli. O executeSystemCli não é suportado e está atualmente disponível apenas para o Data ONTAP que funciona no modo 7D.	Nenhum

Considerações ao converter comandos do PowerShell para Perl

Você deve estar ciente de certas considerações importantes ao converter comandos do PowerShell para Perl porque o PowerShell e o Perl têm recursos diferentes.

Tipos de entrada de comando

O OnCommand Workflow Automation (WFA) permite que os designers de fluxo de trabalho usem arrays e hash como entrada para o comando ao definir um comando. Esses tipos de entrada não podem ser usados quando o comando é definido usando Perl. Se você quiser que um comando Perl aceite entradas de array e hash, você pode definir a entrada como uma string no designer. A definição do comando pode então analisar a entrada, que é passada para criar um array ou hash, conforme necessário. A descrição para a entrada descreve o formato em que a entrada é esperada.

```
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.
```

Declaração do PowerShell

Os exemplos a seguir mostram como uma entrada de array pode ser passada para o PowerShell e Perl. Os exemplos descrevem a entrada CronMonth, que especifica o mês em que a tarefa cron está programada para ser executada. Os valores válidos são inteiros -1 a 11. Um valor de -1 indica que o cronograma é executado a cada mês. Qualquer outro valor denota um mês específico, com 0 sendo janeiro e 11 sendo dezembro.

```
[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,
```

Declaração 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
}
```

Definição do comando

Uma expressão de uma linha no PowerShell usando um operador pipe pode ter que ser expandida em vários blocos de instruções em Perl para alcançar a mesma funcionalidade. Um exemplo de um dos comandos Wait é mostrado na tabela a seguir.

Declaração do PowerShell	Declaração 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)` `}` `}`

Diretrizes para blocos de construção WFA

Você deve estar ciente das diretrizes para o uso de componentes básicos do Workflow Automation.

Diretrizes para SQL no WFA

Você deve estar ciente das diretrizes para usar SQL no OnCommand Workflow Automation (WFA) para escrever consultas SQL para WFA.

SQL é usado nos seguintes locais no WFA:

- Consultas SQL para preencher entradas do usuário para seleção
- Consultas SQL para criar filtros para filtrar objetos de um tipo de entrada de dicionário específico
- Dados estáticos em tabelas no banco de dados playground
- Um tipo de fonte de dados personalizado do tipo SQL onde os dados devem ser extraídos de uma fonte de dados externa, como um banco de dados de gerenciamento de configuração personalizado (CMDB).
- Consultas SQL para scripts de reserva e verificação

Diretrizes	Exemplo
Palavras-chave reservadas SQL devem estar em carateres maiúsculos.	```SELECT vserver.name FROM cm_storage.vserver vserver```
Os nomes de tabela e coluna devem estar em carateres minúsculos.	Tabela: Agregado Coluna: Used_space_mb
Separe palavras com um caractere sublinhado (_). Não são permitidos espaços.	array_performance
O nome da tabela é definido no singular. Uma tabela é uma coleção de uma ou mais entradas.	"função", não "funções"

Diretrizes	Exemplo
Use aliases de tabela com nomes significativos em consultas selecionadas.	``` 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 ```

Diretrizes	Exemplo
Se você tiver que se referir a um parâmetro de entrada de filtro ou parâmetro de entrada de usuário em uma consulta de filtro ou consulta de usuário, use a sintaxe como "inputVariableName'.você também pode usar a sintaxe para se referir a um parâmetro de definição de comando em scripts de reserva e scripts de verificação.	``` 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 ```
Use comentários para consultas complexas. Alguns dos estilos de comentário suportados nas consultas são os seguintes: • "--" até ao final da linha Um espaço é obrigatório após o segundo hífen neste estilo de comentário. • A partir de um caractere "" até o final da linha • A partir de uma sequência "'/*" `to the following` " */"	``` /* 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 ```

Diretrizes para funções WFA

Você pode criar funções para encapsular lógica comumente usada e mais complexa em uma função nomeada e, em seguida, reutilizar a função como valores de parâmetro de comando ou valores de parâmetros de filtro no OnCommand Workflow Automation (WFA).

Diretrizes	Exemplo
Use Camel Case para um nome de função.	CalculateVolumeSize
Nomes de variáveis devem estar em inglês simples e relacionados à funcionalidade da função.	SplitByDelimiter
Não utilize abreviaturas.	CalculateVolumeSize, *not* calcVolSize
As funções são definidas usando MVFLEX Expression Language (MVEL).	Nenhum
A definição da função deve ser especificada de acordo com as diretrizes oficiais da linguagem de Programação Java.	Nenhum

Diretrizes para entradas do dicionário WFA

Você deve estar ciente das diretrizes para criar entradas de dicionário no OnCommand Workflow Automation (WFA).

Diretrizes	Exemplo
Os nomes de entrada do dicionário devem conter apenas carateres alfanuméricos e sublinhados.	Cluster_License Switch_23
Os nomes de entrada do dicionário devem começar com um caractere maiúsculo. Comece cada palavra no nome com um caractere maiúsculo e separe cada palavra com um sublinhado (_).	Volume Array_License
Os nomes dos atributos de entrada do dicionário não devem incluir o nome da entrada do dicionário.	Nenhum
Atributos e referências em uma entrada de dicionário devem ser em carateres minúsculos.	agregado, size_mb
Separe palavras com um sublinhado. Não são permitidos espaços.	resource_pool

Diretrizes	Exemplo
Entradas de dicionário não podem incluir referências que são de um esquema diferente. Quando uma entrada de dicionário requer referência cruzada para um objeto em um esquema diferente, certifique-se de que todas as chaves naturais do objeto a ser referido estão presentes na entrada do dicionário.	A entrada do dicionário Array_Performance requer todas as chaves naturais da entrada do dicionário Array como atributos diretos nela.
Use tipos de dados apropriados para atributos.	Nenhum
Use o tipo de dados longo para atributos relacionados ao tamanho ou ao espaço.	Size_mb e available_size_mb no armazenamento.Entrada do dicionário de volume
Use Enum quando um atributo tiver um conjunto fixo de valores.	raid_type no storage.Entrada do dicionário de volume
Defina "'to be cached'" como true para um atributo ou referência quando uma fonte de dados fornece valor para esse atributo ou referência.para a fonte de dados Active IQ Unified Manager, adicione atributos armazenável em cache se a fonte de dados puder fornecer o valor a ele.	Nenhum
Defina "'can be NULL'" como true se a fonte de dados que fornece o valor para este atributo ou referência pode retornar NULL.	Nenhum
Forneça uma descrição significativa para cada atributo e referência. A descrição é exibida em detalhes de comando ao projetar um fluxo de trabalho.	Nenhum
Não use "'id'" como o nome de um atributo em entradas de dicionário. Ele é reservado para uso interno DO WFA.	Nenhum

Informações relacionadas

Referências ao material de aprendizagem

Diretrizes para comandos

Você deve estar ciente das diretrizes para criar comandos no OnCommand Workflow Automation (WFA).

Diretrizes	Exemplo
Use um nome facilmente identificável para comandos.	`Create Qtree`
Use espaços para delimitar palavras e cada palavra deve começar com um caractere maiúsculo.	`Create Volume`
Forneça uma descrição para explicar a funcionalidade do comando, incluindo o resultado esperado dos parâmetros opcionais.	Nenhum
Por padrão, o tempo limite para comandos padrão é de 600 segundos. O tempo limite padrão é definido durante a criação do comando. Altere o valor padrão somente se o comando puder levar mais tempo para ser concluído.	`Create Volume` comando
No caso de operações de longa duração, crie dois comandos - um para invocar a operação de longa duração e outro para relatar o progresso da operação periodicamente. O primeiro comando deve ser um `Standard Execution` tipo de comando e o segundo deve ser `Wait for Condition` tipo de comando.	`Create VSM` e `Wait for VSM` comandos
Prefix os `Wait for condition` nomes de comando com "'wait'" para fácil identificação.	`Wait for CM Volume Move`
Use um intervalo de espera apropriado para os comandos "'wait for condition'". O valor especificado controla o intervalo no qual o comando de polling é executado para verificar se a operação de longa duração está concluída.	60s intervalo de amostragem para o `Wait for VSM` comando
Para os `Wait for condition` comandos, use um tempo limite apropriado com base no tempo esperado para a operação de longa duração ser concluída. O tempo esperado pode ser consideravelmente mais longo se a operação envolver transferência de dados em uma rede.	Uma transferência de linha de base VSM pode levar muitos dias para ser concluída. Portanto, o tempo limite especificado é de 6 dias.

Representação de cadeia de carateres

A representação de string de um comando exibe os detalhes de um comando em um design de fluxo de trabalho durante o Planejamento e a execução. Somente os parâmetros do comando podem ser usados na representação de string para um comando.

Diretrizes	Exemplo
Evite usar atributos que não tenham nenhum valor. Um atributo sem um valor é exibido como na.	VolName 10.68.66.212[na]aggr1/testVol7
Separe diferentes entradas na representação de string usando os seguintes delimitadores: [ ] , / :	*ArrayName[ArrayIp]*
Forneça rótulos significativos para cada valor na representação de cadeia de carateres.	*Volume name=VolumeName*

Linguagem de definição de comando

Os comandos podem ser escritos usando as seguintes linguagens de script suportadas:

- PowerShell
- Perl

Definição do parâmetro do comando

Os parâmetros do comando são descritos por Nome, Descrição, tipo, um valor padrão para o parâmetro e se o parâmetro é obrigatório. O tipo de parâmetro pode ser String, Boolean, Integer, Long, Double, Enum, DateTime, Capacity, Array, Hashtable, Password ou XmlDocument. Embora os valores para a maioria dos tipos sejam intuitivos, os valores para Array e Hashtable devem estar em um formato específico, conforme descrito na tabela a seguir:

Diretrizes	Exemplo
Certifique-se de que o valor de um tipo de entrada Array seja uma lista de valores, separados por vírgula.	`[parameter(Mandatory=$false, HelpMessage="Months in which the schedule executes.")] [array]$CronMonths` A entrada é passada da seguinte forma: 0,3,6,9
Certifique-se de que o valor de um tipo de entrada Hashtable seja uma lista de pares chave-valor, separados por ponto-e-vírgula.	`[parameter(Mandatory=$false, HelpMessage="Volume names and size (in MB)")] [hashtable]$VolumeNamesAndSize` A entrada é passada da seguinte forma: volume1-100; Volume2-250; Volume3-50

Diretrizes para fluxos de trabalho

Você deve estar ciente das diretrizes para criar ou modificar um fluxo de trabalho predefinido para o OnCommand Workflow Automation (WFA).

Orientações gerais

Diretrizes	Exemplo
Nomeie o fluxo de trabalho de modo que ele reflita a operação executada pelo operador de armazenamento.	`Create a CIFS Share`
Para nomes de fluxo de trabalho, capitalize a letra inicial da primeira palavra e cada palavra que é um objeto. Letras maiúsculas para abreviaturas e acrônimos.	Volume Qtree Crie um compartilhamento CIFS de Qtree Data ONTAP em cluster
Para descrições de fluxo de trabalho, inclua todas as etapas importantes do fluxo de trabalho, incluindo quaisquer pré-requisitos, resultado do fluxo de trabalho ou aspetos condicionais de execução.	Veja a descrição do fluxo de trabalho de amostra `Create VMware NFS Datastore on Clustered Data ONTAP Storage`, que inclui os pré-requisitos.
Defina "'Pronto para produção'" como `true` somente quando o fluxo de trabalho estiver pronto para produção e puder ser exibido na página do portal.	Nenhum
Por padrão, defina "considerar elementos reservados" como verdadeiro. Ao visualizar um fluxo de trabalho para execução, o planejador WFA considera todos os objetos que são reservados junto com os objetos existentes no banco de dados de cache. Efeitos de outros fluxos de trabalho programados ou fluxos de trabalho executados em paralelo são considerados ao Planejar um fluxo de trabalho específico, se essa opção estiver definida como `true`.	• Cenário 1 O fluxo de trabalho 1 cria um volume e está programado para ser executado uma semana depois. O fluxo de trabalho 2 cria qtrees ou LUNs em volumes que são pesquisados e, se o fluxo de trabalho 2 for executado dentro de um dia ou mais, você deve desativar "considerar elementos reservados" para o fluxo de trabalho 2 para evitar que ele considere o volume que deve ser criado em uma semana. • Cenário 2 O fluxo de trabalho 1 usa o `Create Volume` comando. Se houver um fluxo de trabalho programado 2 que consome 100 GB de um agregado, o fluxo de trabalho 1 deve considerar os requisitos do fluxo de trabalho 2 durante o Planejamento.

Diretrizes	Exemplo
Por padrão, ""enable element existence validation"" é definido como `true`.	• Cenário 1 Se você criar um fluxo de trabalho que primeiro remove um volume por nome usando o comando `Remove Volume` somente se o volume existir, e o recriá-lo usando outro comando `Create Volume` como ou `Clone Volume`, então o fluxo de trabalho não deve usar esse sinalizador. O efeito de remover o volume não estará disponível para o `Create volume` comando, fazendo com que o fluxo de trabalho falhe. • Cenário 2 O `Create Volume` comando é usado em um fluxo de trabalho com um nome específico como "'vol198'". Se essa opção estiver definida como true, o planejador DO WFA verifica durante o Planejamento para ver se existe um volume com esse nome na matriz específica. Se o volume existir, o fluxo de trabalho falhará durante o Planejamento.
Quando o mesmo comando for selecionado mais de uma vez em um fluxo de trabalho, forneça nomes de exibição apropriados para as instâncias de comando.	O fluxo de trabalho de exemplo "'criar, mapear e proteger LUNs com SnapVault'" usa o `Create Volume` comando duas vezes. No entanto, ele usa os nomes de exibição como `Create Primary Volume` e `Create Secondary Volume` adequadamente para o volume primário e o volume de destino espelhado.

Entradas do utilizador

Diretrizes	Exemplo
Nomes: • Comece o nome com o caractere "". • Use uma letra maiúscula no início de cada palavra. • Use letras maiúsculas para todos os termos e abreviaturas. • Não utilize sublinhados.	`$Array` `$VolumeName`

Diretrizes	Exemplo
Nomes de exibição: • Use uma letra maiúscula no início de cada palavra. • Separe palavras com espaços. • Se as entradas tiverem unidades específicas, especifique a unidade entre parênteses diretamente no nome do visor.	`Volume Name` `Volume Size (MB)`
Descrições: • Forneça uma descrição significativa para cada entrada de usuário. • Forneça exemplos quando necessário. Você deve fazer isso especialmente quando a entrada do usuário é esperada para estar em um formato específico. As descrições de entrada do usuário são exibidas como dicas de ferramentas para as entradas do usuário durante a execução do fluxo de trabalho.	Iniciadores a serem adicionados a um "iGroup". Por exemplo, IQN ou WWPN do iniciador.
Tipo: Selecione Enum como o tipo se você quiser restringir a entrada a um conjunto específico de valores.	Protocolo: "iscsi", "fcp", "indexado"
Tipo: Selecione consulta como o tipo quando o usuário pode selecionar entre os valores disponíveis no cache WFA.	Tipo DE CONSULTA com consulta da seguinte forma: `SELECT` `ip, name` `FROM` `storage.array`
Tipo: Marque a entrada do usuário como bloqueada quando a entrada do usuário deve ser restrita aos valores obtidos de uma consulta ou deve ser restrita apenas aos tipos Enum suportados.	Tipo de consulta: Somente os storages no cache podem ser selecionados. Protocolo: Tipo de Enum bloqueado com valores válidos como iscsi, fcp, misto. Nenhum outro valor além do valor válido é suportado.
Type: Query TypeAdicione colunas adicionais como valores de retorno na consulta quando ajuda o operador de armazenamento a fazer a escolha certa da entrada do usuário.	Forneça nome, tamanho total, tamanho disponível para que o operador conheça os atributos antes de selecionar o agregado.

Diretrizes	Exemplo
Tipo: Consulta TypeSQL consulta para entradas de usuário pode se referir a quaisquer outras entradas de usuário que a precedem. Isso pode ser usado para limitar os resultados de uma consulta com base em outras entradas do usuário, como unidades do vFiler de um array, volumes de um agregado, LUNs em uma máquina virtual de storage (SVM).	No fluxo de trabalho de exemplo `Create a Clustered Data ONTAP Volume`,, a consulta para VserverName é a seguinte: `SELECT` `  vserver.name` `FROM` `  cm_storage.cluster cluster,` `  cm_storage.vserver vserver` `WHERE` `  vserver.cluster_id = cluster.id` `  AND cluster.name = '${ClusterName}'` `  AND vserver.type = 'cluster'` `ORDER BY` `  vserver.name ASC` A consulta refere-se a "ClusterName", em que "ClusterName" é o nome da entrada do usuário anterior à entrada do usuário VserverName.
Tipo: Use o tipo booleano com valores como "'true, false'" para entradas de usuário que são booleanas por natureza. Isso ajuda a escrever expressões internas no design do fluxo de trabalho usando a entrada do usuário diretamente. Por exemplo, UserInputName em vez de UserInputName.	`$CreateCIFSShare`: Tipo booleano com valores válidos como "verdadeiro" ou "falso"
Tipo:para tipo de cadeia de carateres e número, use expressões regulares na coluna valores quando quiser validar o valor com formatos específicos. Use expressões regulares para entradas de endereço IP e máscara de rede.	A entrada de usuário específica de localização pode ser expressa como "[A-Z][A-Z] 0[1-9]". Esta entrada de usuário aceita valores como "'US-01", "'NB-02", mas não "'nb-00".
Tipo: Para o tipo de número, uma validação baseada em intervalo pode ser especificada na coluna valores.	Para o número de LUNs a serem criados, a entrada na coluna valores é 1-20.
Grupo: Agrupar entradas de usuários relacionadas em intervalos apropriados e nomear o grupo.	"Detalhes de armazenamento" para todas as entradas de usuário relacionadas ao armazenamento. ""Detalhes do datastore"" para todas as entradas de usuário relacionadas ao VMware.

Diretrizes	Exemplo
Obrigatório: Se o valor de qualquer entrada de usuário for necessário para que o fluxo de trabalho seja executado, marque a entrada de usuário como obrigatória. Isso garante que a tela de entrada do usuário aceita obrigatoriamente essa entrada do usuário.	No fluxo de trabalho "Create NFS volume".
Valor padrão: Se uma entrada de usuário tiver um valor padrão que possa funcionar para a maioria das execuções de fluxo de trabalho, forneça os valores. Isso ajuda a permitir que o usuário forneça menos entradas durante a execução, se o padrão serve a finalidade.	Nenhum

Constantes, variáveis e parâmetros de retorno

Diretrizes	Exemplo
Constantes: Defina constantes ao usar um valor comum para definir parâmetros para vários comandos.	*AGREGAÇÃO_COMPROMETIMENTO_THRESHOLD* no `Create, map, and protect LUNs with SnapVault sample workflow`.
Constantes:nomes • Use uma letra maiúscula no início de cada palavra. • Use letras maiúsculas para todos os termos e abreviaturas. • Não utilize sublinhados. • Use letras maiúsculas para todas as letras de nomes constantes.	*AGREEMENT_USED_SPACE_THRESHOLD* *ActualVolumeSizeInMB*
Variáveis: Forneça um nome para um objeto definido em uma das caixas de parâmetro de comando. Variáveis são nomes gerados automaticamente e podem ser alteradas.	Nenhum
Variáveis: Os nomes usam carateres minúsculos para nomes de variáveis.	volume1 partilha_cifs

Parâmetros de retorno: Use os parâmetros de retorno quando o Planejamento e a execução do fluxo de trabalho devem retornar alguns valores calculados ou selecionados durante o Planejamento. Os valores são disponibilizados no modo de visualização quando o fluxo de trabalho é executado a partir de um serviço da Web também.	Agregado: Se o agregado for selecionado usando a lógica de seleção de recursos, o agregado selecionado real pode ser definido como um parâmetro de retorno.

Diretrizes para criar scripts de validação para tipos de sistema remotos

Você deve estar ciente das diretrizes para criar scripts de validação que são usados para testar os tipos de sistema remoto que você define no OnCommand Workflow Automation (WFA).

- O script Perl que você criar deve ser semelhante ao script de exemplo fornecido na janela Script de Validação.
- A saída do script de validação deve ser semelhante à do script de exemplo.

Script de validação de amostra

```
# 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;
}
```

Diretrizes para criar tipos de fonte de dados

Você deve estar ciente das diretrizes para criar tipos de fonte de dados que são usados para definir fontes de dados personalizadas para o OnCommand Workflow Automation (WFA).

Você pode definir um tipo de fonte de dados usando um dos seguintes métodos:

- SQL: Você pode usar as diretrizes WFA SQL para definir consultas selecionadas de fontes de dados com base em um banco de dados externo.
- Script: Você pode escrever um script do PowerShell que fornece os dados para um esquema específico de entradas de dicionário.

As diretrizes para a criação de tipos de fonte de dados são as seguintes:

- Você deve usar a linguagem do PowerShell deve ser usada para criar script.

- O script do PowerShell deve fornecer a saída para cada entrada do dicionário em seu diretório de trabalho atual.
- Os arquivos de dados devem ser nomeados `dictionary_entry.csv`, onde o nome da entrada do dicionário deve ser em carateres minúsculos.

  O tipo de fonte de dados predefinido que coleta informações do Performance Advisor usa um tipo de fonte de dados baseado EM SCRIPT. Os ficheiros de saída têm o nome `array_performance.csv` e. `aggregate_performance.csv`.

- O `.csv` arquivo deve incluir o conteúdo na ordem exata como o dos atributos de entrada do dicionário.

  Uma entrada de dicionário inclui atributos na seguinte ordem: Array_IP, data, dia, hora, CPU_Busy, Total_OPS_per_sec, Disk_throughput_per_sec

  O script do PowerShell adiciona dados ao `.csv` arquivo na mesma ordem.

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

- Você deve usar Encoding para garantir que a saída de dados do script seja carregada no cache DO WFA com precisão.
- Você deve usar N ao inserir um valor nulo no `.csv` arquivo.

Informações sobre direitos autorais

Informações sobre marcas comerciais