WFA のコーディングガイドライン

OnCommand Workflow Automation

NetApp
October 09, 2025

目次

WFA のコーディングガイドライン

フィルタ、機能、コマンド、ワークフローなど、さまざまなビルディングブロックの作成に関する OnCommand Workflow Automation （ WFA ）のコーディングに関する一般的なガイドライン、命名規則、および推奨事項を理解しておく必要があります。

変数のガイドライン

コマンドまたはデータソースの種類を作成するときは、 OnCommand Workflow Automation （ WFA ）で PowerShell 変数と Perl 変数のガイドラインに注意する必要があります。

PowerShell の変数

ガイドライン	例
スクリプト入力パラメータの場合： • Pascal の事例を使用してください。 • アンダースコアは使用しないでください。 • 省略形は使用しないでください。	「 $VolumeName 」 $AutoDeleteOptions' 「 $Size 」
スクリプト内部変数の場合： • Camel case を使用します。 • アンダースコアは使用しないでください。 • 省略形は使用しないでください。	「 $newVolume 」 「 $qtreeName 」 「 $TIME 」
関数の場合： • Pascal の事例を使用してください。 • アンダースコアは使用しないでください。 • 省略形は使用しないでください。	「 GetVolumeSize 」
変数名では大文字と小文字は区別されません。ただし、読みやすくするために、同じ名前に異なる大文字と小文字を使用しないでください。	「 $VARIABLE 」は「 $VARIABLE 」と同じです
変数名は、プレーン英語で記述され、スクリプトの機能に関連した名前にする必要があります。	「 $a`. 」ではなく「 $name` 」を使用してください

ガイドライン	例
各変数のデータ型を明示的に宣言します。	[string] name [int] サイズ
特殊文字 (! @ # & % 、 .）とスペース。	なし
PowerShell の予約キーワードは使用しないでください。	なし
入力パラメータをグループ化するには、まず必須パラメータを配置し、続けてオプションパラメータを配置します。	`param(` `[parameter(Mandatory=$true)]` `[string]$Type,` `[parameter(Mandatory=$true)]` `[string]$Ip,` `[parameter(Mandatory=$false)]` `[string]$VolumeName` `)`
すべての入力変数には '*HelpMessage*' 注釈と意味のあるヘルプ・メッセージを使用してコメントを付けます	`[parameter(Mandatory=$false,HelpMessage="LUN to map")]` `[string]$LUNName`
変数名として「ファイラー」を使用しないでください。代わりに「アレイ」を使用してください。	なし
引数が列挙値を取得する場合は '*ValidateSet*' 注釈を使用しますこれにより、パラメータの Enum データ型に自動的に変換されます。	`[parameter(Mandatory=$false,HelpMessage="Volume state")]` `[ValidateSet("online","offline","restricted")]` `[string]$State`

ガイドライン	例
パラメータの末尾に「 _Capacity 」が付いたエイリアスを追加して、パラメータが容量タイプであることを示します。	「 Create Volume 」コマンドでは、次のようにエイリアスを使用します。 `[parameter(Mandatory=$false,HelpMessage="Volume increment size in MB")]` `[Alias("AutosizeIncrementSize_Capacity")]` `[int]$AutosizeIncrementSize`
パラメータがパスワードタイプであることを示すために、エイリアスを "_Password" で終わるパラメータに追加します。	`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 変数

ガイドライン	例
スクリプト入力パラメータの場合： • Pascal の事例を使用してください。 • アンダースコアは使用しないでください。 • 省略形は使用しないでください。	「 $VolumeName 」 $AutoDeleteOptions' 「 $Size 」
スクリプトの内部変数には省略形を使用しないでください。	$new_volume 「 $qtree_name 」のようになります 「 $TIME 」
関数には省略形を使用しないでください。	'get_volume_size

ガイドライン	例
変数名では大文字と小文字が区別されます。読みやすくするために、同じ名前に異なる大文字と小文字を使用しないでください。	「 $VARIABLE 」は「 $VARIABLE 」と同じではありません
変数名は、プレーン英語で記述され、スクリプトの機能に関連した名前にする必要があります。	「 $a`. 」ではなく「 $name` 」を使用してください
入力パラメータをグループ化するには、まず必須パラメータを配置し、続けてオプションパラメータを配置します。	なし
GetOptions 関数で、入力パラメータの各変数のデータ型を明示的に宣言します。	`GetOptions( "Name=s"=>\$Name, "Size=i"=>\$Size )`
変数名として「ファイラー」を使用しないでください。代わりに「アレイ」を使用してください。	なし
Perl には ' 列挙値の '*ValidateSet*' 注釈は含まれません引数が列挙値を取得する場合は ' 明示的な if 文を使用します	`if (defined$SpaceGuarantee&&!($SpaceGuaranteeeq'none'`
	$SpaceGuaranteeeq’volume'
	$SpaceGuaranteeeq’file')) { die’Illegal SpaceGuarantee argument: \".$SpaceGuarantee.'\"; } ----
すべての Perl WFA コマンドでは、変数、参照、サブルーチンに安全でない構成要素を使用しないようにするために、 "strict" プラグマを使用する必要があります。	`use strict; # the above is equivalent to use strictvars; use strictsubs; use strictrefs;`

ガイドライン	例
すべての Perl WFA コマンドでは、次の Perl モジュールを使用する必要があります。 • getopt これは、入力パラメータの指定に使用されます。 • WFAUtil のようになります コマンドロギング、コマンドの進捗状況の報告、アレイコントローラへの接続などに使用されるユーティリティ機能に使用されます。	`use Getopt::Long;` `use NaServer;` `use WFAUtil;`

インデントのガイドライン

OnCommand Workflow Automation （ WFA ）用の PowerShell または Perl スクリプトを作成する場合は、インデント設定のガイドラインに注意する必要があります。

ガイドライン	例
タブは、 4 つの空白スペースに等しい。	
ブロックの先頭と末尾を表示するには、タブと波かっこを使用します。	PowerShell スクリプト `if` `($pair.length-ne 2)` `{` `throw "Got wrong input data"` `}` Perl スクリプト `if` `(defined $MaxDirectorySize)` `{` `# convert from MBytes to Bytes` `my $MaxDirectorySizeBytes =` `$MaxDirectorySize *` `1024 * 1024;` `}`

ガイドライン	例
オペレーションのセットまたはコードのチャンク間に空白行を追加します。	``` $options=$option.trim(); $pair=$option.split(" "); Get-WFAlogger -Info -messages $("split options: "+ $Pair) ```

コメントのガイドライン

OnCommand Workflow Automation （ WFA ）用スクリプトでの PowerShell と Perl のコメントに関するガイドラインに注意する必要があります。

PowerShell のコメントを表示します

ガイドライン	例
1 行のコメントには # 文字を使用します。	``` # Single line comment $options=$option.trim(); ```
行末のコメントには # 文字を使用します。	``` $options=$option.trim(); # End of line comment ```
ブロックコメントには、との文字を使用します。	``` <# This is a block comment #> $options=$option.trim(); ```

Perl のコメント

ガイドライン	例
1 行のコメントには # 文字を使用します。	`# convert from MBytes to Bytes` `my $MaxDirectorySizeBytes =` `$MaxDirectorySize *` `1024 * 1024;`
行末のコメントには # 文字を使用します。	`my $MaxDirectorySizeBytes =` `$MaxDirect` `orySiZe * 1024 * 1024; # convert` `to Bytes`
先頭と末尾に空白の # を含む各行に # 文字を使用して、複数行コメントのコメント境界を作成します。	`#` `# 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` `#`
WFA コマンドには、コメント化したコードやデッドコードを含めないでください。ただし、テスト目的では、 Plain Old Documentation （ PoD ）メカニズムを使用してコードをコメントアウトできます。	`=begin comment` `# Set deduplication` `if(defined $Deduplication &&` `$Deduplication eq "enabled")` `{` `$wfaUtil-` `>sendLog("Enabling` `Deduplication");` `}` `=end comment` `=cut`

ロギングのガイドライン

OnCommand Workflow Automation （ WFA ）用の PowerShell スクリプトまたは Perl ス

クリプトを作成する際には、ログ記録に関するガイドラインに注意する必要があります。

PowerShell のロギング

ガイドライン	例
ログには Get-WFALogger コマンドレットを使用します。	`Get-WFALogger -Info -message “Creating volume”`
Data ONTAP 、 VMware 、 PowerCLI などの内部パッケージとの対話を必要とするすべてのアクションをログに記録します。すべてのログメッセージは、ワークフローの実行ステータス履歴の実行ログで使用できます。	なし
内部パッケージに渡される関連するすべての引数を記録します。	なし
使用状況に応じて、 Get-WFALogger コマンドレットを使用する場合は、適切なログレベルを使用してください。- INFO 、 - Error 、 - Warn 、 - Debug は、使用可能なさまざまなログレベルです。ログレベルが指定されていない場合、デフォルトのログレベルは Debug です。	なし

Perl のロギング

ガイドライン	例
ログには WFAUtil sendLog を使用します。	`my wfa_util = WFAUtil->new(); eval { $wfa_util->sendLog('INFO', "Connecting to the cluster: $DestinationCluster"); }`
Data ONTAP 、 VMware 、 WFA など、コマンドの外部にある処理とのやり取りが必要なすべてのアクションをログに記録します。WFAUtil sendLog ルーチンを使用して作成するログメッセージは、すべて WFA データベースに格納されます。これらのログメッセージは、実行されるワークフローとコマンドで使用できます。	なし

ガイドライン	例
呼び出されたルーチンに渡されたすべての関連引数を記録します。	なし
適切なログレベルを使用します。 - 情報、 - エラー、 - 警告、 - デバッグは、使用可能なさまざまなログレベルです。	なし
-Info レベルでロギングする場合は、正確かつ簡潔にしてください。ログメッセージにクラス名や関数名などの実装の詳細を指定しないでください。正確な手順またはエラーの正確な説明を英語で入力してください。	次のコードスニペットは、正常なメッセージと不正なメッセージの例を示しています。 `$wfa_util->sendLog('WARN', "Removing volume: '.$VolumeName); # Good Message` `$wfa_util->sendLog('WARN', 'Invoking volume-destroy ZAPI: '.$VolumeName); # Bad message`

エラー処理のガイドライン

OnCommand Workflow Automation （ WFA ）用の PowerShell または Perl スクリプトを作成する際に、エラー処理に関するガイドラインに注意する必要があります。

PowerShell のエラー処理

ガイドライン	例
PowerShell ランタイムによってコマンドレットに追加される共通パラメータには、 ErrorAction や WarningAction などのエラー処理パラメータがあります。 • ErrorAction パラメータは、コマンドレットが終了しないエラーに応答する方法を決定します。 • WarningAction パラメータは、コマンドレットがコマンドレットの警告にどのように対応するかを決定します。 • Stop 、 SilentlyContinue 、 inquire 、 Continue は、 ErrorAction パラメータおよび WarningAction パラメータの有効な値です。 詳細については、 PowerShell CLI の「 Get-Help About_CommonParameters 」コマンドを使用してください。	ErrorAction ：次の例は、終了しないエラーを終了エラーとして処理する方法を示しています。 `New-NcIgroup-Name $IgroupName-Protocol $Protocol-Type$OSType-ErrorActionstop` 警告アクション `New-VM-Name $VMName-VM $SourceVM-DataStore$DataStoreName-VMHost$VMHost-WarningActionSilentlyContinue`
着信例外のタイプが不明な場合は、一般的な "try/catch" ステートメントを使用します。	`try { "In Try/catch block" } catch { "Got exception" }`
着信例外のタイプがわかっている場合は、特定の「 try/catch 」ステートメントを使用します。	`try { "In Try/catch block" } catch[System.Net.WebExceptional], [System.IO. IOException] { "Got exception" }`

ガイドライン	例
「 finally 」文を使ってリソースを解放します。	`try { "In Try/catch block" } catch { "Got exception" } finally { "Release resources" }`
PowerShell の自動変数を使用して、例外に関する情報にアクセスします。	`try { Get-WFALogger -Info -message $("Creating Ipspace: " + $Ipspace) New-NaNetIpspace-Name $Ipspace } catch { Throw "Failed to create Ipspace. Message: " + $_.Exception.Message; }`

Perl エラー処理

ガイドライン	例
Perl には、 try/catch ブロックに対するネイティブ言語サポートは含まれていません。eval ブロックを使用して、エラーの確認と処理を行います。評価ブロックはできるだけ小さくしてください。	``` 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", $@); ```

WFA での PowerShell と Perl の一般的な規則

既存のスクリプトと整合性のあるスクリプトを作成するために、 WFA で使用される PowerShell と Perl の特定の規則を理解しておく必要があります。

- スクリプトで何を実行するかを明確にするために役立つ変数を使用します。
- コメントなしで理解できる読み取り可能なコードを記述してください。
- スクリプトとコマンドはできるだけシンプルにしてください。
- PowerShell スクリプトの場合：
  - 可能なかぎりコマンドレットを使用してください。
  - 使用可能なコマンドレットがない場合は、 .NET コードを呼び出します。
- Perl スクリプトの場合：

- 改行文字を含む "die `" ステートメントは必ず終了してください。

  改行文字が含まれていない場合は、スクリプトの行番号が出力されます。これは、 WFA で実行する Perl コマンドのデバッグには役立ちません。

- 「 getopt 」モジュールで、文字列引数をコマンドに必須にします。

Windows にバンドルされている Perl モジュール

一部の Perl モジュールは、 OnCommand Workflow Automation （ WFA ）用の Windows Active 状態 Perl ディストリビューションにバンドルされています。これらの Perl モジュールは、 Windows に付属している場合にのみ、コマンドの記述に Perl コードで使用できます。

次の表に、 Windows for WFA にバンドルされている Perl データベースモジュールを示します。

データベースモジュール	説明
DBD::mysql	MySQL データベースへの接続を可能にする Perl5 データベースインタフェースドライバ。
試してみましょう	評価ブロックを使用して一般的なミスを最小限に抑えます
XML::libxml	DOM 、 SAX 、 XMLReader インターフェイスを持つ XML および HTML パーサーを提供する libxml2 へのインターフェイス。
DBD ： Cassandra	CQL3 クエリ言語を使用する Cassandra 用の Perl5 データベースインターフェイスドライバ。

カスタムの PowerShell モジュールと Perl モジュールを追加する場合の考慮事項

OnCommand Workflow Automation （ WFA ）に PowerShell および Perl のカスタムモジュールを追加する前に、一定の考慮事項について理解しておく必要があります。カスタムの PowerShell モジュールと Perl モジュールを使用すると、ワークフローを作成するためのカスタムコマンドを使用できます。

- WFA コマンドの実行中に、すべてのカスタム PowerShell モジュールが WFA インストールディレクトリ「 /posh/modules 」に自動的にインポートされます。
- 「 wfa/perl 」ディレクトリに追加されたすべてのカスタム Perl モジュールは、 _@INC_library に含まれています。
- カスタムの PowerShell モジュールと Perl モジュールは、 WFA のバックアップ処理の一環としてバックアップされません。
- カスタムの PowerShell モジュールおよび Perl モジュールは、 WFA のリストア処理中にリストアされま

せん。

新しい WFA にコピーするには、カスタムの PowerShell モジュールと Perl モジュールを手動でバックアップする必要があります。

modules ディレクトリ内のフォルダ名は ' モジュール名と同じである必要があります

WFA のコマンドレットと機能

OnCommand Workflow Automation （ WFA ）には、複数の PowerShell コマンドレットと、 WFA コマンドで使用できる PowerShell および Perl の機能が用意されています。

WFA サーバが提供するすべての PowerShell コマンドレットと機能を表示するには、次の PowerShell コマンドを使用します。

- get-command - モジュール WFAWrapper
- 「 getcommand - Module WFA 」

WFA サーバが提供する Perl の機能は、すべて「 WFAUtil.pm 」モジュールで確認できます。WFA ヘルプモジュールのサポートリンクにあるヘルプセクション、 WFA PowerShell コマンドレットヘルプと WFA Perl メソッドを使用すると、 PowerShell のすべてのコマンドレットと機能、および Perl の機能にアクセスできます。

PowerShell および Perl WFA モジュール

ワークフローのスクリプトを作成するには、 OnCommand Workflow Automation （ WFA ）用の PowerShell または Perl モジュールを理解しておく必要があります。

PowerShell モジュール

ガイドライン	例
Data ONTAP PS Toolkit を使用して、 Toolkit が使用可能になったときにいつでも API を呼び出すことができます。	[Add VLAN] コマンドでは ' 次のようにツールキットを使用します 「 Add-NaNetVlan-Interface $Interface-VLANs $VlanID 」
Data ONTAP PS Toolkit で使用できるコマンドレットがない場合は、「 Invoke-NaSSH 」コマンドを使用して、 Data ONTAP で CLI を呼び出します。	Invoke-NaSsh -Name$ArrayName コマンド「 ifconfig -a 」 -Credential $Credentials 」が実行されます

Perl モジュール

NaServer モジュールは WFA のコマンドで使用されます。NaServer モジュールを使用すると、 Data ONTAP システムのアクティブ管理で使用される Data ONTAP API の呼び出しが可能になります。

ガイドライン	例
NetApp Manageability SDK が使用可能な場合は、NaServer モジュールを使用して API を呼び出します。	以下に、 SnapMirror の再開処理に NaServer モジュールを使用する例を示します。 `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");`

ガイドライン	例
Data ONTAP API が使用できない場合は、executeSystemCli ユーティリティメソッドを使用して Data ONTAP CLI を呼び出します。 executeSystemCli はサポートされておらず、現在は 7-Mode の Data ONTAP でのみ使用できます。	なし

PowerShell コマンドを Perl に変換する際の考慮事項

PowerShell と Perl の機能は異なるため、 PowerShell コマンドを Perl に変換する場合は、いくつかの重要な考慮事項に注意する必要があります。

コマンド入力タイプ

OnCommand Workflow Automation （ WFA ）を使用すると、ワークフローの設計者は、コマンドを定義する際に、コマンドの入力としてアレイとハッシュを使用できます。これらの入力タイプは、 Perl を使用してコマンドを定義する場合には使用できません。Perl コマンドで配列とハッシュの入力を受け入れる場合は、デザイナで文字列として入力を定義できます。コマンド定義では、入力を解析できます。この入力は必要に応じて配列またはハッシュを作成するために渡されます。入力の概要 は、入力が想定される形式です。

```
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 ステートメント

次の例は、 PowerShell および Perl へのアレイ入力の受け渡し方法を示しています。例では、 cron ジョブの実行がスケジュールされている月を指定する、 CronMonth 入力について説明しています。有効な値は、整数 -1 ～ 11 です。値 -1 は、スケジュールが毎月実行されることを示します。その他の値は特定の月を表し、 0 は 1 月、 11 は 12 月を表します。

```
[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 ステートメント

```
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
}
```

コマンド定義

PowerShell でパイプ演算子を使用する 1 行式を、 Perl で複数のステートメントブロックに拡張して、同じ機能を実現しなければならない場合があります。次の表に、待機コマンドの例を示します。

PowerShell ステートメント	**Perl** ステートメント
`# 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)` `}` `}`

WFA のビルディングブロックに関するガイドライン

Workflow Automation ビルディングブロックの使用に関するガイドラインを確認しておく必要があります。

WFA の SQL に関するガイドライン

OnCommand Workflow Automation （ WFA ）で SQL を使用して WFA 用の SQL クエリを記述する際のガイドラインに注意する必要があります。

WFA では、 SQL は次の場所で使用されます。

- 選択のためのユーザ入力を入力する SQL クエリ
- 特定のディクショナリエントリタイプのオブジェクトをフィルタリングするフィルタを作成するための SQL クエリ
- プレイグラウンドデータベース内のテーブル内の静的データ
- カスタム構成管理データベース（ CMDB ）などの外部データソースからデータを抽出する必要がある SQL タイプのカスタムデータソース。
- 予約および検証スクリプトを照会する SQL です

ガイドライン	例
SQL の予約キーワードは大文字で入力する必要があります。	`SELECT vserver.name FROM cm_storage.vserver vserver`
テーブル名と列名は小文字で入力する必要があります。	表：アグリゲート 列： Used_space_MB
単語はアンダースコア（ _ ）文字で区切ります。スペースは使用できません。	array_performance
テーブル名が単数で定義されています。テーブルは 1 つ以上のエントリの集合です。	「関数」ではなく「関数」

ガイドライン	例
SELECT クエリで意味のある名前を持つテーブルエイリアスを使用します	``` 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 ```

ガイドライン	例
フィルタクエリまたはユーザクエリでフィルタ入力パラメータまたはユーザ入力パラメータを参照する必要がある場合は、構文を「 $ ｛inputVariableName ｝ 」として使用します。予約スクリプトおよび検証スクリプトでコマンド定義パラメータを参照する場合は、構文を使用することもできます。	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
複雑なクエリにはコメントを使用します。クエリでサポートされているコメントスタイルの一部を次に示します。 • "`-- 行の最後まで このコメントスタイルでは、 2 番目のハイフンの後にスペースは必要です。 • 行の末尾までの「 # 」文字から • “/” から次の "/`" シーケンスに移動します	/* 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

WFA の機能に関するガイドラインを参照してください

よく使用されるより複雑なロジックを名前付き関数にカプセル化する関数を作成し、その関数をコマンドパラメータ値として再利用したり、 OnCommand Workflow Automation （ WFA ）でパラメータ値をフィルタリングしたりすることができます。

ガイドライン	例
関数名には Camel case を使用します。	計算ボリュームサイズ
変数名は、標準的な英語で、関数の機能に関連している必要があります。	splitByDelimiter
省略形は使用しないでください。	calculateVolumeSize 、 _not_calcVolSize
関数は、 MVEL （ MVEL ）を使用して定義します。	なし
関数定義は、公式の Java プログラミング言語のガイドラインに従って指定する必要があります。	なし

WFA ディクショナリエントリのガイドライン

OnCommand Workflow Automation （ WFA ）でディクショナリエントリを作成するためのガイドラインを確認しておく必要があります。

ガイドライン	例
ディクショナリエントリ名には、英数字とアンダースコアのみを使用する必要があります。	Cluster License （クラスタライセンス） switch_23
辞書エントリ名の先頭は大文字にする必要があります。名前のすべての単語の先頭には大文字を入力し、各単語をアンダースコア（ _ ）で区切ります。	ボリューム array_License
ディクショナリエントリの属性名にディクショナリエントリの名前を含めることはできません。	なし
ディクショナリエントリ内の属性と参照は、小文字で記述する必要があります。	アグリゲート、 size_MB
単語はアンダースコアで区切ります。スペースは使用できません。	resource_pool を指定します

ガイドライン	例
辞書エントリには、別のスキームからの参照を含めることはできません。ディクショナリエントリが別のスキームのオブジェクトへの相互参照を必要とする場合は、参照されるオブジェクトのすべての自然キーがディクショナリエントリに存在することを確認します。	array_Performance ディクショナリエントリでは、Array ディクショナリエントリのすべての自然キーが直接属性として必要になります。
属性に適切なデータ型を使用します。	なし
サイズやスペースに関連する属性には、長いデータ型を使用します。	storage.Volume 辞書エントリの size_mb と available_size MB です
属性の値が固定されている場合は、 Enum を使用します。	storage.1 の raid_type 。ボリューム辞書エントリ
データソースがその属性または参照の値を提供する場合は、属性または参照の「キャッシュする」を true に設定します。 Active IQ Unified Manager データソースの場合、データソースがその属性に値を提供できる場合は、キャッシュ可能な属性を追加します。	なし
この属性または参照の値を提供するデータソースが NULL を返す可能性がある場合は `` を null にすることができます	なし
各属性と参照に意味のある概要 を指定します。ワークフローを設計する際には、コマンドの詳細に概要が表示されます。	なし
ディクショナリエントリ内の属性の名前として「 id` 」を使用しないでください。 WFA の内部使用のために予約されています。	なし

- 関連情報 *

学習資料への参照

コマンドのガイドライン

OnCommand Workflow Automation （ WFA ）でコマンドを作成する際のガイドラインに注意する必要があります。

ガイドライン	例
コマンドには、わかりやすい名前を使用します。	qtree を作成します

ガイドライン	例
単語を区切るにはスペースを使用します。各単語は大文字で始まる必要があります。	「ボリュームを作成」
オプションパラメータで想定される結果など、コマンドの機能について説明する概要 を提供します。	なし
デフォルトでは、標準コマンドのタイムアウトは 600 秒です。コマンドの作成時にデフォルトのタイムアウトが設定されます。デフォルト値を変更するのは、コマンドの完了に時間がかかる場合だけにしてください。	[Create Volume] コマンドを使用します
長時間の処理の場合は、 2 つのコマンドを作成します。 1 つは長時間の処理を実行するコマンドで、もう 1 つは処理の進捗状況を定期的に報告するコマンドです。最初のコマンドは「標準実行」コマンドタイプで、 2 番目のコマンドは「条件を待機」コマンドタイプである必要があります。	'Create VSM' および 'Wait for VSM' コマンド
容易に識別できるように 'wait for condition' コマンド名の前には 'wait' を付けます	「 CM ボリューム移動の待機」
「条件の待機」コマンドに適切な待機間隔を使用します。指定した値は、長時間の処理が完了したかどうかを確認するためにポーリングコマンドが実行される間隔を制御します。	'Wait for VSM' コマンドの 60 秒のサンプリング間隔
「条件待機」コマンドでは、長時間実行動作が完了するまでの予想時間に基づいて適切なタイムアウトを使用します。ネットワーク経由のデータ転送を実行する場合は、想定時間が大幅に長くなる可能性があります。	VSM ベースライン転送が完了するまでに数日かかることがあります。したがって、指定されたタイムアウトは 6 日です。

文字列表現

コマンドの文字列表現は、計画および実行中にワークフロー設計内のコマンドの詳細を表示します。コマンドの文字列表現で使用できるのは、コマンドパラメータだけです。

ガイドライン	例
値のない属性は使用しないでください。値のない属性は NA と表示されます。	volname 10.68.66.212 [NA]aggr1 / testVol7
文字列表現では、 [],/ の区切り記号を使用して異なるエントリを区切ります。	*`ArrayName [ArrayIp]`*

ガイドライン	例
文字列表現のすべての値に意味のあるラベルを指定します。	*Volume name=VolumeName*

コマンド定義言語

コマンドは、次のサポートされているスクリプト言語を使用して記述できます。

- PowerShell
- Perl の場合

コマンドパラメータの定義

コマンドパラメータは、名前（ Name ）、パラメータ（概要）、タイプ（ Type ）、パラメータのデフォルト値、およびパラメータが必須かどうかによって記述されます。パラメータタイプは、 String 、 Boolean 、 Integer 、 Long 、 Double 、 列挙、日時、容量、アレイ、ハッシュテーブル、 パスワード、または XmlDocument 。ほとんどの型の値は直感的ですが、 Array と Hashtable の値は次の表に示す特定の形式にする必要があります。

ガイドライン	例
Array 入力タイプの値がカンマで区切った値のリストであることを確認します。	`[parameter(Mandatory=$false, HelpMessage="Months in which the schedule executes.")] [array]$CronMonths` 入力は 0 、 3 、 6 、 9 のように渡されます
Hashtable 入力型の値が、セミコロンで区切られた key= 値のペアのリストであることを確認します。	`[parameter(Mandatory=$false, HelpMessage="Volume names and size (in MB)")] [hashtable]$VolumeNamesAndSize` 指定するパスは、次のとおりです。ボリューム 1 = 100 、ボリューム 2 = 250 、ボリューム 3 = 50

ワークフローのガイドライン

OnCommand Workflow Automation （ WFA ）の事前定義されたワークフローの作成または変更に関するガイドラインを確認しておく必要があります。

一般的なガイドライン

ガイドライン	例
ストレージオペレータが実行した処理が反映されるように、ワークフローに名前を付けます。	CIFS 共有を作成します
ワークフロー名では、最初の単語の最初の文字とオブジェクトであるすべての単語を大文字にします。略語や頭字語の文字を大文字にします。	ボリューム qtree clustered Data ONTAP の qtree CIFS 共有を作成
ワークフローの説明には、ワークフローの前提条件、ワークフローの結果、または条件付きの実行など、ワークフローのすべての重要な手順を含めます。	前提条件を含むサンプルワークフロー「 clustered Data ONTAP ストレージでの VMware NFS データストアの作成」の概要 を参照してください。
ワークフローがプロダクションの準備ができていて、ポータルページに表示できる場合にのみ、「 Ready for Production 」を「 true 」に設定します。	なし
デフォルトでは ' 予約済み要素の考慮を true に設定します実行用のワークフローをプレビューするとき、 WFA プランナーは、キャッシュデータベース内の既存のオブジェクトと一緒に予約されているすべてのオブジェクトを考慮します。このオプションを「 true 」に設定すると、特定のワークフローを計画するときに、他のスケジュールされたワークフローまたは並列実行ワークフローの影響が考慮されます	• シナリオ 1 ワークフロー 1 ではボリュームを作成し、 1 週間後に実行するようにスケジュール設定します。ワークフロー 2 では、検索対象のボリュームに qtree または LUN が作成されます。ワークフロー 2 が 1 日以内に実行される場合、ワークフロー 2 で「予約済み要素の考慮」をオフにして、 1 週間に作成するボリュームを考慮しないようにする必要があります。 • シナリオ 2 ワークフロー 1 では 'Create Volume' コマンドを使用しますスケジュール設定されたワークフロー 2 でアグリゲートの 100GB を消費する場合は、計画段階でワークフロー 2 の要件を考慮する必要があります。

ガイドライン	例
デフォルトでは、「エレメントの存在検証を有効にする」は「真」に設定されています	• シナリオ 1 ボリュームが存在する場合にのみ ' ボリュームを削除コマンドを使用してボリュームを名前で削除するワークフローを作成し ' そのボリュームを作成ボリュームまたはクローンボリュームなどの別のコマンドを使用して再作成する場合 ' ワークフローではこのフラグを使用しないでくださいボリュームを削除した場合の効果は 'Create volume コマンドでは使用できないため ' ワークフローは失敗します • シナリオ 2 Create Volume コマンドは ' vol198` という名前のワークフローで使用されます このオプションが true に設定されている場合、WFA プランナーは、計画時に、その名前を使用してボリュームが特定のアレイに存在するかどうかを確認します。計画中にボリュームが存在するとワークフローが失敗します。
ワークフローで同じコマンドを複数回選択する場合は、コマンドインスタンスに適切な表示名を指定します。	サンプル・ワークフローでは 'Create Volume' コマンドを 2 回使用して 'LUN の作成 ' マッピング ' 保護を SnapVault で行いますただし ' プライマリ・ボリュームとミラー・デスティネーション・ボリュームには ' プライマリ・ボリュームの作成とセカンダリ・ボリュームの作成が適切に使用されます

ユーザ入力

ガイドライン	例
名前： • 名前の先頭には「 $` 」文字を付けます。 • 各単語の先頭に大文字を使用します。 • すべての用語や略語に大文字を使用します。 • アンダースコアは使用しないでください。	「 $Array 」 「 $VolumeName 」

ガイドライン	例
表示名： • 各単語の先頭に大文字を使用します。 • 単語をスペースで区切ります。 • 入力に特定の単位がある場合は、表示名に角かっこで囲んで単位を指定します。	「ボリューム名」 「ボリューム・サイズ（ MB ）」
説明： • ユーザ入力ごとにわかりやすい概要 を指定します。 • 必要に応じて例を挙げてください。 これは、特にユーザ入力が特定の形式であると想定される場合に実行してください。 ユーザー入力の説明は、ワークフローの実行中にユーザー入力のツールチップとして表示されます。	「 igroup 」に追加するイニシエータたとえば、イニシエータの IQN や WWPN などです。
Type ：入力を特定の値セットに制限する場合は、タイプとして Enum を選択します。	プロトコル : "iSCSI`", "FCP `", "MIXED `"
Type ：ユーザが WFA キャッシュの値から選択できる場合は、タイプとして Query を選択します。	$Array: クエリのクエリタイプ : `SELECT` `ip, name` `FROM` `storage.array`
[タイプ]: ユーザー入力がクエリから取得した値に制限されているか、サポートされている列挙型のみに制限されている必要がある場合に、ユーザー入力をロック済みとしてマークします。	$Array ：ロックされたクエリタイプ：キャッシュ内のアレイのみ選択できます。 $Protocol ：有効な値が iSCSI 、 FCP 、 mixed のロックされた Enum タイプです。有効な値以外の値はサポートされません。
タイプ：クエリタイプクエリ演算子がユーザ入力を適切に選択できるようになると、クエリに戻り値として列を追加できます。	$ Aggregate ：アグリゲートを選択する前に属性を確認できるように、名前と合計サイズ、使用可能なサイズを指定します。

ガイドライン	例
タイプ：クエリータイプユーザ入力の SQL クエリーは、その前にある他のユーザ入力を参照できます。この機能を使用すると、アレイの vFiler ユニット、アグリゲートのボリューム、 Storage Virtual Machine （ SVM ）の LUN など、他のユーザ入力に基づいてクエリの結果を制限できます。	サンプル・ワークフローでは 'Create a Clustered Data ONTAP Volume] で 'VserverName のクエリは次のようになります `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` クエリは $ \ ｛ clustername ｝ を参照します。 $clustername は、 $VserverName ユーザ入力の前に入力されたユーザ名です。
type ：ブール型を使用し、ブール型の値を "true 、 false" として使用します。これにより、ユーザ入力を直接使用して、ワークフロー設計で内部式を記述できます。たとえば、 $UserInputName='Yes' ではなく $UserInputName とします。	$CreateCIFSShare: 有効な値が「 true 」または「 false 」のブール型
[タイプ]: 文字列および数値型の場合、特定の形式で値を検証するときは、 [値] 列で正規表現を使用します。 IP アドレスとネットワークマスクの入力には正規表現を使用します。	場所に固有のユーザ入力は、「 [A-Z][A-Z]\-0[1-9]` 」と表現できます。このユーザー入力は "US -01`" 、 "NB-02`" などの値を受け入れますが、 "nb-00`" は受け入れません。
[タイプ]: 数値タイプの場合、 [値] 列で範囲ベースの検証を指定できます。	作成する LUN の数については、「値」列のエントリは 1~20 です。
グループ：グループに関連するユーザが該当するバケットに入力し、グループに名前を付けます。	ストレージ関連のすべてのユーザー入力用の「ストレージの詳細」。VMware 関連のすべてのユーザー入力の「データストアの詳細」。

ガイドライン	例
必須：ワークフローを実行するためにユーザ入力の値が必要な場合は、ユーザ入力を必須としてマークします。これにより、ユーザ入力画面がユーザからの入力を受け入れられるようになります。	「 Create NFS Volume 」ワークフローの「 $VolumeName 」
デフォルト値：ユーザ入力にデフォルト値があり、ほとんどのワークフロー実行で有効な場合は、値を指定します。これにより、デフォルトで目的が達成された場合に、実行中に入力を減らすことができます。	なし

定数、変数、および戻りパラメータ

ガイドライン	例
定数 : 複数のコマンドにパラメータを定義するために共通の値を使用する場合は、定数を定義します。	SnapVault サンプル・ワークフローでの LUN の作成 ' マッピング ' 保護については *'aggregate_O*オーバーコミットメント *_threshold* を参照してください
定数 : 名前 • 各単語の先頭に大文字を使用します。 • すべての用語や略語に大文字を使用します。 • アンダースコアは使用しないでください。 • 定数名のすべての文字に大文字を使用します。	*aggregate_Used_space_threshold* *ActualVolumeSizeInMB*
変数：コマンドパラメータのいずれかのボックスで定義されたオブジェクトに名前を指定します。変数は自動的に生成される名前で、変更できます。	なし
変数：変数名には小文字を使用します。	ボリューム 1 cifs_share
戻りパラメータ：ワークフローの計画と実行で、計画中に計算値または選択した値が返される場合は、戻りパラメータを使用します。ワークフローが Web サービスから実行されたときにも、プレビューモードで値が使用可能になります。	アグリゲート：リソース選択ロジックを使用してアグリゲートを選択した場合、選択した実際のアグリゲートを戻りパラメータとして定義できます。

リモートシステムタイプの検証スクリプトを作成する際のガイドライン

OnCommand Workflow Automation （ WFA ）で定義したリモートシステムタイプをテストするための検証スクリプトを作成する際のガイドラインを確認しておく必要があります。

- 作成する Perl スクリプトは、 [検証スクリプト] ウィンドウに表示されるサンプルスクリプトに似ている必要があります。
- 検証スクリプトの出力は、サンプルスクリプトの出力と同様である必要があります。

サンプルの検証スクリプト

```
# 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;
}
```

データソースタイプの作成に関するガイドライン

OnCommand Workflow Automation （ WFA ）のカスタムデータソースの定義に使用するデータソースタイプの作成に関するガイドラインを確認しておく必要があります。

データソースタイプは、次のいずれかの方法で定義できます。

- SQL ： WFA の SQL ガイドラインを使用して、外部データベースに基づいてデータソースからの選択クエリを定義できます。
- スクリプト：特定のディクショナリ方式にデータを提供する PowerShell スクリプトを記述できます。

データソースタイプの作成に関するガイドラインは次のとおりです。

- スクリプトの作成には PowerShell の言語を使用する必要があります。
- PowerShell スクリプトは、現在の作業ディレクトリ内の各ディクショナリエントリの出力を提供する必要があります。
- データ・ファイルには 'dictionary_entry_csv' という名前を付ける必要がありますこの場合 ' ディクショナリ・エントリの名前は小文字にする必要があります

  Performance Advisor から情報を収集する事前定義されたデータソースのタイプでは、スクリプトベースのデータソースのタイプを使用します。出力ファイルの名前は 'array_performe.csv' および 'aggregate_performe.csv' です

- 「 .csv 」ファイルには、ディクショナリエントリ属性と同じ順序でコンテンツが含まれている必要があります。

  ディクショナリエントリには、 array_ip 、 date 、 day 、 hour 、 cpu_busy 、 total_ops_per_sec 、 disk_per_sec

  PowerShell スクリプトは '.csv ファイルに同じ順序でデータを追加します

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

- スクリプトからのデータ出力が WFA キャッシュに正確にロードされるようにするには、エンコーディングを使用します。
- “ .csv ”ファイルにヌル値を入力するときは、“ N ”を使用する必要があります。

著作権に関する情報

商標に関する情報