Create a command

OnCommand Workflow Automation 5.0

NetApp
April 19, 2024

Table of Contents

Create a command . . . 1
Test the reservation script for commands . . . 3

Create a command

You can create a WFA command to complete a specific task in your workflow if there is no predefined WFA command that is suitable for the task.

What you’ll need

You must know how to write the required code for the command using either PowerShell or Perl.

Steps

1. Click **Designer** > **Commands**.
2. Click  on the toolbar.
3. In the **Properties** tab of the **New Command Definition** dialog box, enter or select the required details in the **Name**, **Description**, and **Timeout** fields.
   a. In the **String Representation** field, enter a string representation for the command using MVEL syntax.

      Example

      ```
      VolumeName + "=>" + SnapshotName
      ```

      The string representation for a command is used to display the information you want to see in the workflow design during planning and execution. You must use only the parameters of the command in string representation for a command.

   b. **Optional:** If you are creating a wait command, select **Wait for condition** in the **Command type** section, and set the required value in the **Waiting interval (s)** field.
4. In the **Code** tab, perform the following steps:
   a. Select the required scripting language for the command from the **Script Language** list.

      You can click + and select an additional language for the command.

   b. Enter the appropriate code for the command in the selected language tab.

      If you want to use password type for user inputs in the PowerShell script, you must create an alias for the parameter and include `_Password` in the attribute. For Perl script, you can specify the type as **Password** in the **Parameters Definition** tab.

      Example

      ```
      param (
        [parameter(Mandatory=$false, HelpMessage="Specify an AD
      administrator password.")]
        [Alias("ADAdminPassword_Password")]  [string]$ADAdminPassword
      )
      ```

5. In the **Parameters Definition** tab, perform the following steps:
   a. Click **Discover Parameters** to populate the parameters definition table.

      The parameters and their attributes are extracted from the code and displayed in the table. For example, the `Array` and `VolumeName` parameters are extracted from the following code:

```
param (
  [parameter(Mandatory=$true, HelpMessage="Array name or IP
address")]
  [string]$Array,

  [parameter(Mandatory=$true, HelpMessage="Volume name")]
  [string]$VolumeName,
)
```

b. Click the description column of the parameters to edit the description.

   You cannot edit any other field in this tab.

6. In the **Parameters Mapping** tab, perform the following steps for each parameter:

   a. From the **Type** column, select the appropriate dictionary object.

   b. In the **Attribute** column, enter or select the appropriate attributes for the dictionary object from the list.

      After entering an attribute, you can enter a period (.) and include another attribute of that object.

      Example

      Enter `cm_storage.volume` as type and `aggregate.name` as the attribute for the `AggregateName` parameter.

   c. In the **Object Name** column, enter an object name.

      The object name is used for grouping the parameters under a tab in the Parameters for <command> dialog box when you are specifying the command details in a workflow.

   The unmapped parameters are displayed in the **Other parameters** tab of the Parameters for <command> dialog box when you are specifying the command details in a workflow.

7. **Optional:** In the **Reservation** tab, enter a reservation script using SQL queries to reserve the resources that are required by the command during a scheduled workflow execution:

   a. In the **Reservation Representation** field, enter a string representation for the reservation using MVEL syntax.

      Example

      `"Add rule for SnapMirror label "` + SnapMirrorLabel + `" to the SnapMirror policy "` + PolicyName + "`"`

      The string representation is used to display the details of the resources reserved in the Reservations window.

      The reservation script must not perform any operation on databases except the cm_storage, cm_performance, storage, performance, vc, and custom schemes.

8. **Optional:** In the **Verification** tab, enter an SQL query to verify whether the command has affected the data sources and the WFA cache as expected so that the reservation can be removed.

   The SQL query that you enter can only consist of SQL SELECT statements.

   a. Click **Test Verification** to test the verification script.
   b. In the **Verification** dialog box, enter the required test parameters.
   c. If you do not want to use the reservation data to test the verification script, clear the **Use reservation data in test** field.
   d. Click **Test**.
   e. After reviewing the test result, close the dialog box.
9. Click **Test** to test the command.
10. In the Testing Command <command name> dialog box, click **Test**.

    The result of the test is displayed in the Log messages section of the dialog box.

11. Click **Save**.

Test the reservation script for commands

You can test the reservation scripts you have written for OnCommand Workflow Automation (WFA) commands on the playground database to ensure that the scripts are working fine and not affecting the WFA database tables.

About this task

The default WFA installation path is used in this procedure. If you changed the default location during installation, you must use the changed WFA installation path.

Steps

1. Open a command prompt on the WFA server and change directories to the following location: `c:\Program Files\NetApp\WFA\mysql\bin`
2. Create a dump of the WFA database using the following command:

   ```
   mysqldump -u wfa -pWfa123 --single-transaction --skip-add-drop-table
   database_tables> dump_location
   ```

   Example

   Command to create a dump of the cm_storage database tables:

   ```
   mysqldump -u wfa -pWfa123 --single-transaction --skip-add-drop-table
   cm_storage> c:\tmp\cmSt2.sql
   ```

3. Restore the dump you have created on to the WFA playground database using the following command:

   ```
   mysql -u wfa -pWfa123 playground < dump_location
   ```

   Example

```
mysql -u wfa -pWfa123 playground < c:\tmp\cmSt2.sql
```

4. Create or edit a WFA command and write the reservation script in the **Reservation** tab.

   You must ensure that the reservation and verification scripts use only the playground database.

5. Create or edit a workflow, include the command in the workflow, and then execute the workflow.
6. Verify that the reservation and verification scripts are working as expected.

   The WFA data source acquisition process does not update the playground database. You must remove the reservations created by the command manually.