PDTool Server Module User Guide

An Open Source Asset for use with TIBCO® Data Virtualization

TIBCO Software empowers executives, developers, and business users with Fast Data solutions that make the right data available in real time for faster answers, better decisions, and smarter action. Over the past 15 years, thousands of businesses across the globe have relied on TIBCO technology to integrate their applications and ecosystems, analyze their data, and create real-time solutions. Learn how TIBCO turns data—big or small—into differentiation at www.tibco.com.

Project Name	AS Assets PDTool (Promotion and Deployment Tool)
Document Location	This document is only valid on the day it was printed. The source of the document will be found in the PDTool and PDToolRelease folder (https://github.com/TIBCOSoftware)
Purpose	User's Guide

www.tibco.com

Global Headquarters
3303 Hillview Avenue
Palo Alto, CA 94304

Tel: +1 650-846-1000
+1 800-420-8450
Fax: +1 650-846-1005

Revision History

Version	Date	Author	Comments
1.0	8/11/2011	Gordon Rose	Initial revision for Server Manager Module User Guide
3.0	8/21/2013	Mike Tinius	Updated docs to Cisco format.
3.1	2/18/2014	Mike Tinius	Prepare docs for open source.
3.2	11/17/2014	Mike Tinius	Update license.
3.3	3/4/2015	Mike Tinius	Updated table of contents to include methods and updated docs to Cisco format.
4.0	12/14/2017	Mike Tinius	Initial revision with Tibco
5.0	08/27/2020	Mike Tinius	Updated documentation
5.1	10/20/2020	Mike Tinius	Updated documentation

Related Documents

Name	Author
PDTool User's Guide.pdf	Mike Tinius

Supported Versions

Name	Version
TIBCO® Data Virtualization	7.0.8 or later

Table of Contents

1 Introduction

Purpose

The purpose of the Server Manager Module User Guide is to demonstrate how to effectively use the Server Manager Module and execute actions. The Server Manager Module performs the tasks of starting, stopping and restarting Data Virtualization (DV).

Audience

This document is intended to provide guidance for the following users:

- Architects
- Developers
- Administrators
- Operations personnel

References

Product references are shown below. Any references to CIS or DV refer to the current TIBCO® Data Virtualization.

- TIBCO® Data Virtualization was formerly known as
    - Cisco Data Virtualization (DV)
    - Composite Information Server (CIS)

2 Server Manager Module Definition

Method Definitions and Signatures

1. startServer

startServer communicates over the network with an instance of the CIS monitor process to remotely start CIS.

```
@param serverId target server id from servers config xml
@param pathToServersXML path to the server values xml
@throws CompositeException
public void startServer(String serverId, String pathToServersXML) throws
CompositeException;
```

2. stopServer

stopServer communicates over the network with an instance of the CIS monitor process to remotely start CIS.

```
@param serverId target server id from servers config xml
@param pathToServersXML path to the server values xml
@throws CompositeException
public void stopServer(String serverId, String pathToServersXML) throws
CompositeException;
```

3. restartServer

restartServer communicates over the network with an instance of the CIS monitor process to remotely start CIS.

```
@param serverId target server id from servers config xml
@param pathToServersXML path to the server values xml
@throws CompositeException
public void restartServer(String serverId, String pathToServersXML)
throws CompositeException;
```

General Notes:

The argument pathToServersXML will be located in [PDTool/resources/modules). The value passed into the methods will be the fully qualified path. The paths get resolved when executing the property file and evaluating the $MODULE_HOME variable.

3 Server Manager Module XML Configuration

A full description of the PDTool Module XML Schema can be found by reviewing /docs/PDToolModules.xsd.html.

Description of the Module XML

The Server Manager Module performs CIS start, stop and restart operations. The Server Manager Module does not depend on a module-specific XML file. Only the servers.xml file, which is common to all modules, is used by this module.

4 How To Execute

The following section describes how to setup a property file for both command line and Ant and execute the script.

Script Execution

The full details on property file setup and script execution can be found in the document "*PDTool User's Guide.pdf*". The abridged version is as follows:

```
Windows: ExecutePDTool.bat -exec ../resources/plans/UnitTest-ServerManager.dp

Unix: ./ExecutePDTool.sh -exec ../resources/plans/UnitTest- ServerManager.dp
```

Properties File (UnitTest-ServerManager.dp):

Property File Rules:

```
# ----------------------------
# UnitTest-ServerManager.dp
# ----------------------------
#   1. All parameters are space separated.  Commas are not used.
#         a. Any number of spaces may occur before or after any parameter and are
#            trimmed.
#
#   2. Parameters should always be enclosed in double quotes according to these rules:
#         a. when the parameter value contains a comma separated list:
#                                    ANSWER: "ds1,ds2,ds3"
#
#         b. when the parameter value contain spaces or contains a dynamic variable
#            that will resolve to spaces
#            i.   There is no distinguishing between Windows and Unix variables.
#                 Both UNIX style variables ($VAR) and
#                 and Windows style variables (%VAR%) are valid and will be parsed
#                 accordingly.
#            ii.  All parameters that need to be grouped together that contain spaces
#                 are enclosed in double quotes.
#            iii. All paths that contain or will resolve to a space must be enclosed
#                 in double quotes.
#                 An environment variable (e.g. $MODULE_HOME) gets resolved on
#                 invocation PDTool.
#                       Paths containing spaces must be enclosed in double quotes:
#                              ANSWER: "$MODULE_HOME/LabVCSModule.xml"
#                       Given that
#                       MODULE_HOME=C:/dev/Cis Deploy Tool/resources/modules
#                       , PDTool automatically resolves the variable to
#                       "C:/dev/Cis Deploy Tool/resources/modules/LabVCSModule.xml".
#
```

```
#         c. when the parameter value is complex and the inner value contains spaces
#                     i. In this example $PROJECT_HOME will resolve to a path that
#                        contains spaces such as C:/dev/Cis Deploy Tool
#                        For example take the parameter
#                        -pkgfile $PROJECT_HOME$/bin/carfiles/testout.car.
#                        Since the entire command contains a space it must be enclosed
#                        in double quotes:
#                               ANSWER:
#                        "-pkgfile $PROJECT_HOME/bin/carfiles/testout.car"
#
#   3. A comment is designated by a # sign preceding any other text.
#         a. Comments may occur on any line and will not be processed.
#
#   4. Blank lines are not processed
#         a. Blank lines are counted as lines for display purposes
#         b. If the last line of the file is blank, it is not counted for display
#            purposes.
#
```

Property File Parameters:

```
# ----------------------------
# Parameter Specification:
# ----------------------------
# Param1=[PASS or FAIL]  :: Expected Regression Behavior.  Informs the script whether
# you expect the action to pass or fail.  Can be used for regression testing.
# Param2=[TRUE or FALSE] :: Exit Orchestration script on error
# Param3=Module Batch/Shell Script name to execute (no extension).  Extension is added
# by script.
# Param4=Module Action to execute
# Param5-ParamN=Specific space separated parameters for the action.  See Property
# Rules below.
```

Property File Example:

```
# ----------------------------------------
# Begin task definition list:
# ----------------------------------------
# Quick Test
PASS   FALSE   ExecuteAction   stopServer     $SERVERID  "$MODULE_HOME/servers.xml"
PASS   FALSE   ExecuteAction   startServer    $SERVERID  "$MODULE_HOME/servers.xml"
PASS   FALSE   ExecuteAction   restartServer  $SERVERID  "$MODULE_HOME/servers.xml"
```

Ant Execution

The full details on build file setup and ant execution can be found in the document "*PDTool User's Guide.pdf*". The abridged version is as follows:

```
Windows: ExecutePDTool.bat -ant ../resources/ant/build-ServerManager.xml

Unix: ./ExecutePDTool.sh -ant ../resources/ant/build-ServerManager.xml
```

Build File:

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="PDTool" default="default" basedir=".">

  <description>Server Manager Module Ant script</description>

  <!-- Default properties -->
  <property name="SERVERID"                  value="localhost"/>
  <property name="noarguments"               value="&amp;quot;&amp;quot;"/>

  <!-- Default Path properties -->
  <property name="RESOURCE_HOME"             value="${PROJECT_HOME}/resources"/>
  <property name="MODULE_HOME"               value="${RESOURCE_HOME}/modules"/>
  <property name="pathToServersXML"          value="${MODULE_HOME}/servers.xml"/>
  <property name="pathToArchiveXML"          value="${MODULE_HOME}/ArchiveModule.xml"/>
  <property name="pathToDataSourcesXML"      value="${MODULE_HOME}/DataSourceModule.xml"/>
  <property name="pathToGroupsXML"           value="${MODULE_HOME}/GroupModule.xml"/>
  <property name="pathToPrivilegeXML"        value="${MODULE_HOME}/PrivilegeModule.xml"/>
  <property name="pathToRebindXML"           value="${MODULE_HOME}/RebindModule.xml"/>
  <property name="pathToRegressionXML"       value="${MODULE_HOME}/RegressionModule.xml"/>
  <property name="pathToResourceXML"         value="${MODULE_HOME}/ResourceModule.xml"/>
  <property name="pathToResourceCacheXML"    value="${MODULE_HOME}/ResourceCacheModule.xml"/>
  <property name="pathToServerAttributeXML"  value="${MODULE_HOME}/ServerAttributeModule.xml"/>
  <property name="pathToTriggerXML"          value="${MODULE_HOME}/TriggerModule.xml"/>
  <property name="pathToUsersXML"            value="${MODULE_HOME}/UserModule.xml"/>
  <property name="pathToVCSModuleXML"        value="${MODULE_HOME}/VCSModule.xml"/>

  <!-- Custom properties -->
  Place your property names here:
  <property name="moduleIds"                 value="srv1,srv2"/>

  <!-- Default Classpath [Do Not Change] -->
  <path id="project.class.path">
        <fileset dir="${PROJECT_HOME}/lib"><include name="**/*.jar"/></fileset>
        <fileset dir="${PROJECT_HOME}/dist"><include name="**/*.jar"/></fileset>
        <fileset dir="${PROJECT_HOME}/ext/ant/lib"><include name="**/*.jar"/></fileset>
  </path>

  <taskdef name="executeJavaAction" description="Execute Java Action"
classname="com.tibco.ps.deploytool.ant.CompositeAntTask" classpathref="project.class.path"/>

  <!-- ================================
       target: default
    ================================ -->
  <target name="default" description="Restart CIS">

<executeJavaAction description="restartServer" action="restartServer"
arguments="$(moduleIds)^${pathToServersXML}" endExecutionOnTaskFailure="TRUE"/>
</target>
</project>
```

Module ID Usage

The following explanation provides a general pattern for module identifiers. The module identifier for this module is “serverIds”.

- Possible values for the module identifier:
- 1. ***Inclusion List*** - CSV string like “id1,id2”
    - PDTool will process only the passed in identifiers in the specified module XML file.

Example command-line property file

```
PASS   FALSE  ExecuteAction startServer "svr1,svr2" "$MODULE_HOME/servers.xml"
```

Example Ant build file

```
<executeJavaAction description="startServer" action="startServer"
arguments="svr1,svr2^${pathToServersXML}" endExecutionOnTaskFailure="TRUE"/>
```

- 2. ***Process All*** - '*' or whatever is configured to indicate all resources
    - PDTool will process all resources in the specified module XML file.

Example command-line property file

```
PASS   FALSE  ExecuteAction  startServer "*" "$MODULE_HOME/servers.xml"
```

Example Ant build file

```
<executeJavaAction description="startServer"   action=" startServer "
arguments="*^${pathToServersXML}" endExecutionOnTaskFailure="TRUE"/>"
```

- 3. ***Exclusion List*** - CSV string with '-' or whatever is configured to indicate exclude resources as prefix like “-id1,id2”
    - PDTool will ignore passed in resources and process the rest of the identifiers in the module XML file.

Example command-line property file

```
PASS   FALSE  ExecuteAction startServer "-svr1,svr2" "$MODULE_HOME/servers.xml"
```

Example Ant build file

```
<executeJavaAction description="startServer" action="startServer" arguments="-
svr1,svr2^${pathToServersXML}" endExecutionOnTaskFailure="TRUE"/>
```

5 PDTool Examples

As the Server Manager Module does not depend on or require any module-specific XML files, please see the sections above for complete examples of how to use the module's startServer, stopServer and restartServer operations.

6 Exceptions and Messages

The following are common exceptions and messages that may occur.

Wrong Number of Arguments:

This may occur when you do not place double quotes around comma separated lists.

7 Conclusion

Concluding Remarks

The Promotion and Deployment Tool is a set of pre-built modules intended to provide a turn-key experience for promoting DV resources from one DV instance to another. The user only requires system administration skills to operate and support. The code is transparent to operations engineers resulting in better supportability. It is easy for users to swap in different implementations of a module using the Spring framework and configuration files.

How you can help!

Build a module and donate the code back to Professional Services for the advancement of the "***Promotion and Deployment Tool***".