PDTool Group 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	6/10/2011	Mike Tinius	Initial revision for Group Module User Guide
1.0.1	8/1/2011	Mike Tinius	Revisions due to Architecture changes
1.2	10/1/2012	Mike Tinius	Fixed doc issue with privilege list
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	3/24/2014	Mike Tinius	Changed references of XML namespace to www.dvbu.cisco.com
3.3	11/17/2014	Mike Tinius	Update license.
3.4	3/4/2015	Mike Tinius	Updated table of contents to include methods and updated docs to Cisco format.
4.	12/14/2017	Mike Tinius	Initial revision with Tibco

Related Documents

Name	Author
PDTool User's Guide.pdf	Mike Tinius

Supported Versions

Name	Version
TIBCO® Data Virtualization	7.0.4 or later

Table of Contents

1 Introduction

Purpose

The purpose of the Group Module User Guide is to demonstrate how to effectively use the Group Module and execute actions. Groups are managed within the browser-based Data Virtualization (DV) Manager. The Group Module will allow the automation of creating, updating, deleting groups and generating the Group Module property file.

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 Group Definition Module

Method Definitions and Signatures

1. createOrUpdateGroups

Create CIS groups. If they already exist, update them instead.

```
@param serverId - target server id from servers config xml
@param groupIds - comma separated list of group Ids
@param pathToGroupsXML - path to the groups xml
@param pathToServersXML - path to the server values xml
@return void
@throws CompositeException


public void createOrUpdateGroups(String serverId, String groupIds, String
pathToGroupsXML, String pathToServersXML) throws CompositeException;
```

2. deleteGroups

Delete CIS groups from a specified domain.

```
@param serverId - target server id from servers config xml
@param groupIds - comma separated list of group Ids
@param pathToGroupsXML - path to the groups xml
@param pathToServersXML - path to the server values xml
@return void
@throws CompositeException


public void deleteGroups(String serverId, String groupIds, String
pathToGroupsXML, String pathToServersXML) throws CompositeException;
```

3. addUsersToGroups

Add passed in users to the associated with group ids and target server Id.

```
@param serverId - target server id from servers config xml
@param groupIds - comma separated list of group Ids
@param userNames - comma separated user names
@param pathToGroupsXML - path to the groups xml
@param pathToServersXML - path to the server values xml
@return void
```

```
@throws CompositeException


public void addUsersToGroups(String serverId, String groupIds, String
deleteUsersFromGroups String userNames,String pathToGroupsXML, String
pathToServersXML) throws CompositeException;
```

4. deleteUsersFromGroups

Delete passed in users from the associated group ids and target server Id.

```
@param serverId - target server id from servers config xml

@param groupIds - comma separated list of group Ids

@param userNames - comma separated user names like username1,username2

@param pathToGroupsXML - path to the groups xml

@param pathToServersXML - path to the server values xml

@return void

@throws CompositeException


public void deleteUsersFromGroups(String serverId, String groupIds,
String userNames,String pathToGroupsXML, String pathToServersXML) throws
CompositeException;
```

5. generateGroupsXML

Export existing CIS groups to a XML file based on the list of passed in group ids and server id.

```
@param serverId - target server id from servers config xml

@param domain - domain name.  If domain is not passed then all groups are
included

@param pathToGroupsXML - path including name to the groups xml which
needs to be created

@param pathToServersXML - path to the server values xml

@return void

@throws CompositeException


public void generateGroupsXML(String serverId,String domainName,String
pathToGroupsXML, String pathToServersXML) throws CompositeException;
```

General Notes:

The arguments pathToGroupsXML and 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 Group Module XML Configuration

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

Description of the Module XML

The GroupModule XML provides a structure “group” for “create, update, delete, manage users” and generating the user XML. The global entry point node is called “GroupModule” and contains zero or more “group” nodes.

```
<?xml version="1.0"?>
<p1:GroupModule xmlns:p1="http://www.dvbu.cisco.com/ps/deploytool/modules">
    <group>
        <id>group1</id>
        <groupName>group1</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>ACCESS_TOOLS</privilege>
    </group>
    <group>
        <id>group2</id>
        <groupName>group2</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>ACCESS_TOOLS MODIFY_ALL_CONFIG MODIFY_ALL_RESOURCES MODIFY_ALL_STATUS
MODIFY_ALL_USERS READ_ALL_CONFIG READ_ALL_RESOURCES READ_ALL_STATUS READ_ALL_USERS
UNLOCK_RESOURCE</privilege>
    </group>
</p1:GroupModule>
```

Attributes of Interest

id – a unique identifier within the file.

groupName – this value is tells the system the name of the group.

groupDomain – this value is tells the system which “valid’ domain the user belongs to.

Attribute Value Restrictions

privilege – A space separated list of Privilege Access Rights that may include 1 or more of [ACCESS_TOOLS MODIFY_ALL_CONFIG MODIFY_ALL_RESOURCES MODIFY_ALL_STATUS MODIFY_ALL_USERS READ_ALL_CONFIG READ_ALL_RESOURCES READ_ALL_STATUS READ_ALL_USERS UNLOCK_RESOURCE]

Schema validation uses the following set:

```
<xs:element name="privilege" maxOccurs="unbounded" minOccurs="0">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="ACCESS_TOOLS"/>
            <xs:enumeration value="MODIFY_ALL_CONFIG"/>
            <xs:enumeration value="MODIFY_ALL_RESOURCES"/>
            <xs:enumeration value="MODIFY_ALL_STATUS"/>
            <xs:enumeration value="MODIFY_ALL_USERS"/>
```

```
                <xs:enumeration value="READ_ALL_CONFIG"/>
                <xs:enumeration value="READ_ALL_RESOURCES"/>
                <xs:enumeration value="READ_ALL_STATUS"/>
                <xs:enumeration value="READ_ALL_USERS"/>
                <xs:enumeration value="UNLOCK_RESOURCE"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
```

4 How To Execute

The following section describes how to setup a property file for both command line and Ant and execute the script. This script will use the GroupModule.xml that was described in the previous section.

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-Group.dp
Unix: ./ExecutePDTool.sh -exec ../resources/plans/UnitTest-Group.dp
Properties File (UnitTest-Group.dp):
Property File Rules:

```
# ----------------------------
# UnitTest-Group.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 for UNIX:
# ----------------------------------------
#
PASS   FALSE  ExecuteAction generateGroupsXML          $SERVERID composite
       $MODULE_HOME/getGroupModule.xml $MODULE_HOME/servers.xml
PASS   FALSE  ExecuteAction   deleteGroups             $SERVERID group1
       $MODULE_HOME/GroupModule.xml $MODULE_HOME/servers.xml
PASS   FALSE  ExecuteAction   createOrUpdateGroups     $SERVERID "group1,group2"
       $MODULE_HOME/GroupModule.xml $MODULE_HOME/servers.xml
PASS   FALSE  ExecuteAction addUsersToGroups           $SERVERID group1 "user3"
       $MODULE_HOME/GroupModule.xml $MODULE_HOME/servers.xml
PASS   FALSE  ExecuteAction deleteUsersFromGroups      $SERVERID group1 "user3"
       $MODULE_HOME/GroupModule.xml $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-Group.xml
Unix: ./ExecutePDTool.sh -ant ../resources/ant/build-Group.xml

Build File:

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

```
  <description>description</description>

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

  <!-- Custom properties -->
  <property name="groupIds"                  value="group1,group2"/>
  <property name="pathToGenGroupXML"         value="${MODULE_HOME}/getGroupModule.xml"/>

  <!-- 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"/>

  <!-- 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="Update CIS with environment specific parameters">
  <!-- Windows / UNIX -->

  <executeJavaAction description="Generate"         action="generateGroupsXML"
        arguments="${SERVERID}^composite^${pathToGenGroupXML}^${pathToServersXML}"
        endExecutionOnTaskFailure="TRUE" endExecutionOnScriptLaunch="TRUE"/>


  <executeJavaAction description="Delete"           action="deleteGroups"
        arguments="${SERVERID}^${groupIds}^${pathToGroupsXML}^${pathToServersXML}"
        endExecutionOnTaskFailure="TRUE" endExecutionOnScriptLaunch="TRUE"/>


  <executeJavaAction description="CreateOrUpdate" action="createOrUpdateGroups"
        arguments="${SERVERID}^${groupIds}^${pathToGroupsXML}^${pathToServersXML}"
        endExecutionOnTaskFailure="TRUE" endExecutionOnScriptLaunch="TRUE"/>
```

```
  <executeJavaAction description="AddUsers"          action="addUsersToGroups"
       arguments="${SERVERID}^${groupIds}^user3^${pathToGroupsXML}^${pathToServersXML}"
       endExecutionOnTaskFailure="TRUE" endExecutionOnScriptLaunch="TRUE"/>


  <executeJavaAction description="DeleteUsers"       action="deleteUsersFromGroups"
       arguments="${SERVERID}^${groupIds}^user3^${pathToGroupsXML}^${pathToServersXML}"
       endExecutionOnTaskFailure="TRUE" endExecutionOnScriptLaunch="TRUE"/>
  </target>
</project>
```

5 PDTool Examples

The following are common scenarios when using the GroupModule.

Scenario 1 – Generate Group XML

Description:

Generate the group xml property file based on the domain “composite”.

XML Configuration Sample:

Not applicable for this example.

Execution Sample:

Unix: ./ExecutePDTool.sh -exec ../resources/plans/UnitTest-Group.dp
Property file setup for UnitTest-Group.dp:

```
# ----------------------------------------
# Begin task definition list for UNIX:
# ----------------------------------------
# Generate
PASS   FALSE  ExecuteAction generateGroupsXML   $SERVERID composite
       $MODULE_HOME/getGroupModule.xml $MODULE_HOME/servers.xml
```

Results Expected:

The file getGroupModule.xml is produced with only groups from the “composite” domain.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:GroupModule xmlns:ns2="http://www.dvbu.cisco.com/ps/deploytool/modules">
    <group>
        <id>admin-0</id>
        <groupName>admin</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>ACCESS_TOOLS MODIFY_ALL_CONFIG MODIFY_ALL_RESOURCES
MODIFY_ALL_STATUS MODIFY_ALL_USERS READ_ALL_CONFIG READ_ALL_RESOURCES
READ_ALL_STATUS READ_ALL_USERS UNLOCK_RESOURCE</privilege>
    </group>
    <group>
        <id>all-1</id>
        <groupName>all</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
    <group>
        <id>group1-2</id>
        <groupName>group1</groupName>
```

```
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
    <group>
        <id>group2-3</id>
        <groupName>group2</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
</ns2:GroupModule>
```

Scenario 2 – Delete Groups

Description:

Delete groups. If the group does not exist then no action is taken.

XML Configuration Sample:

Use the GroupModule XML file and make sure it has an entry that looks like this:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:GroupModule xmlns:ns2="http://www.dvbu.cisco.com/ps/deploytool/modules">
    <group>
        <id>group1</id>
        <groupName>group1</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
    <group>
        <id>group2</id>
        <groupName>group2</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
</ns2:GroupModule>
```

Execution Sample:

Unix: ./ExecutePDTool.sh -exec ../resources/plans/UnitTest-Group.dp
Property file setup for UnitTest-Group.dp:

```
# ----------------------------------------
# Begin task definition list for UNIX:
# ----------------------------------------
# Delete
PASS   FALSE  ExecuteAction   deleteGroups     $SERVERID "group1,group2"
       $MODULE_HOME/GroupModule.xml $MODULE_HOME/servers.xml
```

Results Expected:

The script will report “PASS” for the execution of this action. Open DV Manager and review the list of groups. The groups “group1 and group2” should not exist.

Scenario 3 – Create Or Update Groups

Description:

Create or update groups. If the group does not exist then create it otherwise update it.

XML Configuration Sample:

Use the GroupModule XML file and make sure it has an entry that looks like this:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:GroupModule xmlns:ns2="http://www.dvbu.cisco.com/ps/deploytool/modules">
    <group>
        <id>group1</id>
        <groupName>group1</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
    <group>
        <id>group2</id>
        <groupName>group2</groupName>
        <groupDomain>composite</groupDomain>
        <privilege>NONE</privilege>
    </group>
</ns2:GroupModule>
```

Execution Sample:

Unix: ./ExecutePDTool.sh -exec ../resources/plans/UnitTest-Group.dp
Property file setup for UnitTest-Group.dp:

```
# ----------------------------------------
# Begin task definition list for UNIX:
# ----------------------------------------
# Create or Update
PASS   FALSE  ExecuteAction   createOrUpdateGroups    $SERVERID "group1,group2"
       $MODULE_HOME/GroupModule.xml $MODULE_HOME/servers.xml
```

Results Expected:

The script will report “PASS” for the execution of this action. Open DV Manager and review the list of groups. The groups “group1 and group2” should exist now.

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 DEV 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***".