Database Device Server User's Guide (V2.2)

A.Götz 05/03/2001

This guide explains how user's access the TANGO Database device server. It contains a brief description of how the Database device server works. The main part of the guide consists of a detailed description of all commands implemented and how to call them. Some examples are given and a list of known problems and how to solve them.

Contents

1 Introduction

The TANGO database device server provides a TANGO Device interface to an underlying database. The underlying database is mysql. The device server is a client of the mysql database server. The TANGO tables are accessed using embedded SQL in the Database device server. The client uses the TANGO Device command_inout method to send requests to the Database device server. The present version of the device server is monothreaded. This means simultanous requests to the database are queued. This can be changed if performance becomes a bottleneck.

2 General

There is only one general command for the moment (DbInfo).

2.1 DbInfo

- *description- returns list of general information about database e.g. time database has been running, number of device servers registered and exported, number of hosts, number of clients, number of property etc.*
- *input= void - nothing*
- *returns= Tango::DevString Array* - sequence of strings containing information when database was started, and how many servers, device, attributes and properties are defined in the database e.g.*

```
Running since 2001-03-05 15:03:48

Devices defined = 142
Devices exported = 36
Device servers defined = 60
Device servers exported = 15

Class properties defined = 9
Device properties defined = 147
Class attribute properties defined = 20
Device attribute properties defined = 74
```

- *exception(s) = "DB_SQLError"*

3 Server

A server refers to a TANGO device server. It is a process which runs on Unix or Windows and serves TANGO devices to clients. Every TANGO device server has a unique name (made up of process name/instance name). Every server has a special TANGO device associated with it (DServer/process/instance) which is used to manage the server process remotely. The server related commands support getting the list of servers per host, adding a server with its devices, and unexporting all devices for a server.

3.1 DbGetServerList

- *description- returns list of device server names which match the wild card*
- *input= Tango::DevVarString- wildcard (* matches any character e.g. a*)*
- *returns= Tango::DevVarString Array* - list of fully qualified server names (e.g. steppermotor/id112_01, fluids/sr, ...)*
- *exception(s)= "DB_SQLError"*

3.2 DbGetHostList

- *description- returns list of host names which match the wild card*
- *input= Tango::DevString- wildcard (* matches any character e.g. a*)*
- *returns= Tango::DevVarStringArray* - list of hosts(e.g. id112, corvus, dumela, ...)*
- *exception(s)= "DB_SQLError"*

3.3 DbGetHostServerList

- *description- returns list of servers running on the specified host*
- *input= Tango::DevString- host (e.g. dumela)*
- *returns= Tango::DevVarStringArray* - list of servers (e.g. fluids/opc, ...)*
- *exception(s)= "DB_SQLError"*

3.4 DbAddServer

- *description- add a server with a list of devices to the TANGO database.This command will also add the DServer device for the server name to the database.*

  NOTE: if any of the devices exist already in the database they are first deleted and then reinserted with all dynamic values set to "nada" or 0 and marked as unexported. To re-export the device(s) belonging this server restart the server.

- *input= Tango::DevVarStringArray* - server name followed by a series of device names and classes e.g. "OregonMaxeds/id11", "id11/oms/1", "OregonMaxe", "id11/oms/2", "OregonMaxe", "id11/oms/3", "OregonMaxe"*

- *returns= void - nothing*
- *exception(s)= "DB_SQLError", "DB_IncorrectArguments", "DB_IncorrectDeviceName"*

3.5 DbUnExportServer

- *description- un-export all devices belonging to this server from the database so that clients cannot access it*
- *input= Tango::DevString- server name e.g. "steppermotor/id11"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError"*

3.6 DbDeleteServer

- *description- delete a server and its devices from the database. The device resources are NOT deleted.*
- *input= Tango::DevString- server name e.g. "steppermotor/id11"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError", "DB_IncorrectServerName"*

3.7 DbGetServerInfo

- *description- get information about the server from the database. The command returns the server host, mode, and level.*
- *input= Tango::DevString- server name e.g. "steppermotor/id11"*
- *returns= Tango::DevVarStringArray - host, mode and level e.g. "dumela", "1", "99"*
- *exception(s)= "DB_SQLError"*

3.8 DbPutServerInfo

- *description- put information about the server into the database. The command expects the server name, host, mode, and level.*
- *input= Tango::DevString- server name, host, mode, and level e.g. "steppermotor/id11", "dumela", "1", "99"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError", "DB_IncorrectArguments"*

3.9 DbDeleteServerInfo

- *description- delete information about the server from the database. The command expects the server name.*
- *input= Tango::DevString- server name e.g. "steppermotor/id11"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError"*

4 Class

Every TANGO device belongs to a class. The class implements the device functionality. There are few purely class related commands. Check the properties section for class properties commands.

4.1 DbGetClassList

- *description- returns the list of classes to be served by the specified device server*
- *input= Tango::DevString- device server personal name as e.g." StepperMotor/id11"*
- *returns= Tango::DevVarStringArray* - sequence of device classes e.g. "DServer", "StepperMotor"*
- *exception(s)= "DB_SQLError"*

5 Device

Devices are the distributed control objects of every TANGO control system. They are defined in the database with (or without) their properties.

5.1 DbGetDeviceList

- *description- returns the list of devices to be served by the specified device server for the specified device class*
- *input= Tango::DevVarStringArray* - device server personal name as e.g. "StepperMotor/id11", "StepperMotor"*
- *returns= Tango::DevVarStringArray* - sequence of devices to serve e.g. "id11/motor/1", "id11/motor/2", .*
- *exception(s)= "DB_SQLError", "DB_IncorrectArguments"*

5.2 DbGetDeviceAliasList

- *description- returns list of device alias names which match the wild card*
- *input= Tango::DevString- wildcard (* matches any character e.g. a*)*
- *returns= Tango::DevVarString Array* - list of aliases (tth, om, srct, fluids)*
- *exception(s)= "DB_SQLError"*

5.3 DbGetDeviceMemberList

- *description- returns list of device member names which match the wild card*
- *input= Tango::DevString- wildcard (* matches any character e.g. d/f/m*)*
- *returns= Tango::DevVarStringArray* - list of members (e.g. 01, 02, 03, ...)*
- *exception= "DB_SQLError"*

5.4 DbGetDeviceFamilyList

- *description- returns list of device family names which match the wild card*
- *input= Tango::DevString- wildcard (* matches any character e.g. d/f*)*
- *returns= Tango::DevVarStringArray* - list of families (e.g. motor, d-ct, ...)*
- *exception(s)= "DB_SQLError"*

5.5 DbGetDeviceDomainList

- *description- returns list of device domain names which match the wild card*
- *input= Tango::DevString- wildcard (* matches any character e.g. d*)*
- *returns= Tango::DevVarStringArray* - list of domains (e.g. id11, cryo, sr, ...)*
- *exception(s)= "DB_SQLError"*

5.6 DbGetDeviceClassList

- *description- returns the list of devices and their classes to be served by the specified device server*
- *input= Tango::DevString- device server personal name as e.g." StepperMotor/id11"*
- *returns= Tango::DevVarStringArray* - sequence of device + device class e.g. "id11/motor/1", "StepperMotor", "id11/motor/1", "StepperMotor", . . .*
- *exception(s)= "DB_SQLError"*

5.7 DbGetDeviceExportedList

- *description- returns the list of exported devices whose names satisfy the filter specified on input*
- *input= Tango::DevString* - filter for device names e.g. "*" or "Dserver/*"*
- *returns= Tango::DevVarStringArray* - sequence of device names e.g. "id11/motor/1", "id11/motor/2", "id11/motor/3", ...*
- *exception(s)= "DB_SQLError"*

5.8 DbExportDevice

- *description- export device to database so that clients can access it*
- *input= Tango::DevVarStringArray* - device name followed IOR, host, server name, pid, protocol version number e.g. "id11/motor/1", , "IOR...", "crate031", "StepperMotor/id11", "123", "1"*
- *returns= void - nothing*
- *exception(s)- "DB_SQLError", "DB_IncorrectArguments"*

5.9 DbUnExportDevice

- *description- un-export device to database so that clients cannot access it*
- *input= Tango::DevString- device name e.g. "id11/motor/1"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError"*

5.10 DbImportDevice

- *description- returns device servant information so that client can build up a connection to it*
- *input= Tango::DevString- device name*
- *returns= Tango::DevVarLongStringArray* - string branch contains device name, IOR and TANGO protocol version number, long branch contains 1 if device exported and 0 if device is defined but not exported e.g. svalue[] = "id11/motor/1", "IOR...", "1" and lvalue[] = "1"*
- *exception(s)= "DB_SQLError", "DB_DeviceNotDefined"*

5.11 DbImportDeviceList

- *description- returns a list of device servant information so that client can build up a connection to it*
- *input= Tango::DevVarStringArray* - list of device names e.g. "id11/motor/1", "id99/dummy/device"*
- *returns= Tango::DevVarLongStringArray* - string branch contains device names, IORs and TANGO protocol version numbers long branch contains 1 if device exported , 0 if device is defined but not exported and -1 if device is not defined e.g. svalue[] = "id11/motor/1", "IOR...", "1", "id99/dummy/device", "", "-1" and lvalue[] = "1", "-1"*
- *exception(s)= "DB_SQLError"*

5.12 DbAddDevice

- *description- adds a device to the TANGO database. This command will also add the DServer device for the server name to the database.*

  NOTE: if the device exists already in the database it is first deleted and then reinserted with all dynamic values set to "nada" or 0 and marked as unexported. To re-export the device(s) belonging this server restart the server.
- *input= Tango::DevVarStringArray* - server name followed by device name and class e.g. "Oregon-Maxeds/id11", "id11/oms/1", "OregonMaxe"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError", "DB_IncorrectArguments", "DB_IncorrectDeviceName"*

5.13 DbPutDeviceAlias

- *description- create or change a device's alias*
- *input= Tango::DevString - device name followed by alias name e.g. "id11/motor/1", , "2theta"*
- *returns= void - nothing*
- *exception(s)= "DB_SQLError", "DB_DeviceNotDefined*

5.14 DbDeleteDevice

- *description- deletes a device, its attributes and all its properties from the TANGO database.*
- *input= Tango::DevString - device name e.g. "id11/oms/1"*
- *returns= void - nothing*
- *exception(s)- "DB_SQLError", "DB_IncorrectDeviceName"*

6 Attributes

Attributes are normalised scalar, vector or 2D data types implemented in a device. Attributes can be read or read/write. They have a fixed set of properties e.g. minimum, maximum, unit, type, format, description etc. NOTE: attributes for classes have been also defined but their utility has not been proven (yet). They might be dropped in the future BE WARNED.

6.1 DbGetClassAttributeList

- *description- queries the database for the list of attributes defined for a class. Returns all attributes defined for the class.*
- *input= Tango::DevVarStringArray - class name, wildcard e.g. "OregonMaxe", "*"*
- *returns= Tango::DevVarStringArray* - list of attributes e.g. "position", "acceleration", velocity*
- *exception(s)- "DB_SQLError"*

6.2 DbGetClassAttributeProperty

- *description- returns the list of attribute properties to query for the specified class. All properties for a class attribute are returned. Only simple types are supported i.e. no sequences of simple types.*
- *input= Tango::DevVarStringArray* - class name followed by a list of attributes to query e.g. "OregonMaxe", "position"*
- *returns= Tango::DevVarStringArray* - class name followed by number of attributes queried followed by a list of attributes, number of properties and a list of properties and values e.g. "OregonMaxe", "1", "position", "3" "min", "-25000", "max", "25000", "units", "steps"*
- *exception(s)- "DB_SQLError"*