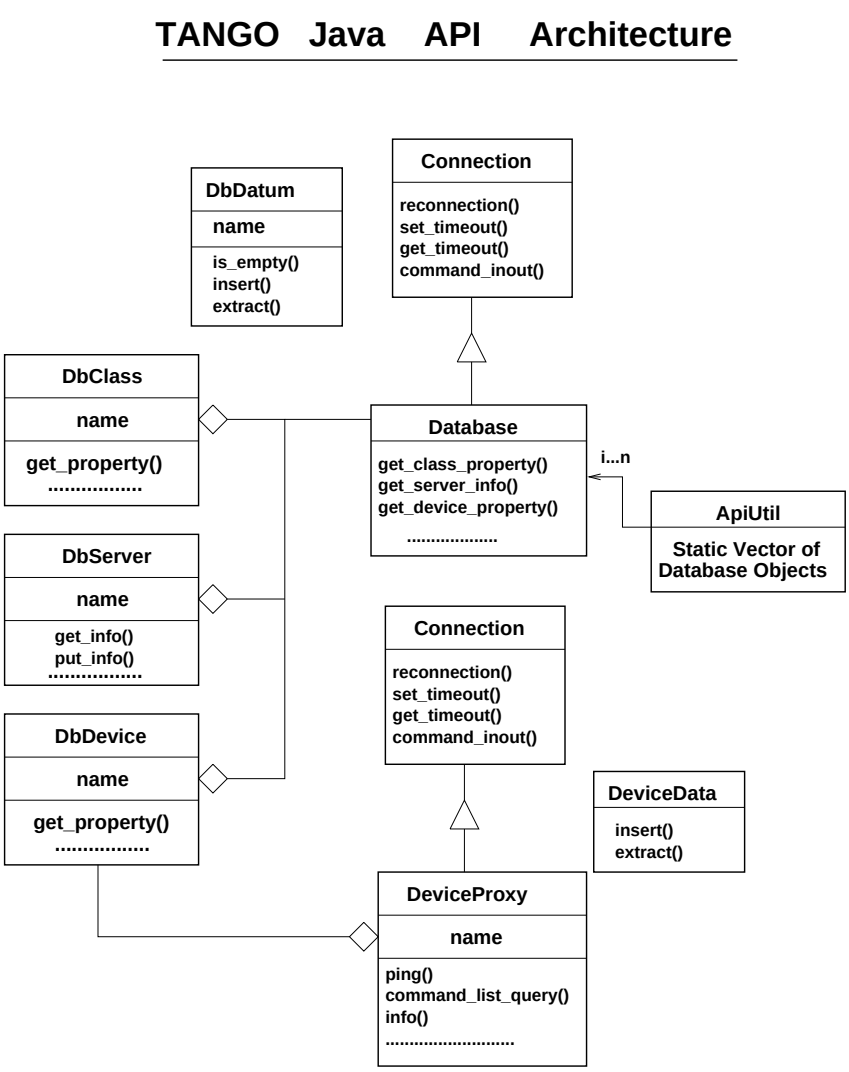
TANGO Java API

Pascal VERDIER

revision 1.3 - 2 April 2001

This paper documents the Java API for the Tango 2 database and device servers.



Contents

Part I
Introduction

1 Description

The TANGO database is implemented today as a TANGO device server. To access it or a device, the user has the CORBA interface command_inout(). This expects and returns all parameters as ascii strings thereby making the database laborious to use for retrieving device properties and information. In order to simplify this access a high-level api has been implemented which hides the low-level formatting necessary to convert the cmd_inout() return values into binary values and all CORBA aspects of the TANGO. All data types are native Java data types e.g. simple type or array. This note documents this high level interface for Java.

2 Basic Philosophy

The basic philosophy is to have high level classes for the database, properties, device and class info in the database and a class for sending and receiving database or device values.

All classes and data types are defined in *fr.esrf.TangoApi* package.

3 Classes

3.1 Data object classes

DeviceData: Obect used to send and receive data on device.

DbDatum: Object used to put or get properties on database.

DbDevInfo: Object used to read device information on database.

DbDevImportInfo: Object used to read imported device information on database.

DbDevExportInfo:Object used to read exported device information on database.

3.2 Devices and Database access classes

Database: Direct access to TANGO database.

DbClass: Class properties access to TANGO database.

DbServer: Server properties access to TANGO database.

DbDevice Device properties access to TANGO database.

DeviceProxy: Device access for commands (extending DbDevice class).

4 Reporting errors

For the device and database classes, most of methods throw an *DevFailed* exception in case of error. See *TANGO Device Server Programmer's Guide Manual* chapter *Reporting Errors* (5.4) , except those which specified.

In opposite, for the data oject classes only the specified method throw *DevFailed* exception in case of error.

5 Getting Started

The quickest way of getting started is by studying this example:

```
/**
 *   Example of a client using the TANGO Api
 */
import fr.esrf.Tango.*;
import fr.esrf.TangoDs.*;
import fr.esrf.TangoApi.*;

public class TestDatabase
{
  public static void main (String args[])
  {
    try
    {
// Database Management.
//-------------------------

// Create a Database object or retrieve an existing connection
          Database  dbase = ApiUtil.get_db_obj();

// Get and display database info.
          System.out.println(dbase.get_info());

// Build a DbDevInfo object (name, class, server)
// to add a device into the database .
          String devname = "tango/admin/corvus";
          DbDevInfo devinfo = new DbDevInfo(devname, "Starter", "Starter/corvus");
          dbase.add_device(devinfo);

// Get and isplay info about device import.
          DbDevImportInfo       imp_info = dbase.import_device(devname);
          System.out.println(imp_info);

// Update device properties.
          devname = "my/serial/device";
          DbDatum[] prop;
          prop = new DbDatum[3];
          prop[0] = new DbDatum("baudrate", 19200);
          prop[1] = new DbDatum("parity",   "none");
          prop[2] = new DbDatum("stopbits", 1);
          dbase.put_property(devname, prop);

// Query the database for device properties.
          long   baud   = 9600;
          String parity = none;
          short  stop   = 1;
          String[]  propnames = { "baudrate", "parity", "stopbits"};
          prop = dbase.get_device_property(devname, propnames);
          if (prop[0].is_empty()==false)  baud   = prop[0].extractLong();
          if (prop[1].is_empty()==false)  parity = prop[1].extractString();
```

```
            if (prop[2].is_empty()==false)  stop   = prop[2].extractShort();


    // Device Management
    //-------------------------

    // get device properties as from database.
            DeviceProxy       dev = new DeviceProxy("my/serial/device");
            prop = dev.get_property(propnames);
            if (prop[0].is_empty()==false)  baud   = prop[0].extractLong();
            if (prop[1].is_empty()==false)  parity = prop[1].extractString();
            if (prop[2].is_empty()==false)  stop   = prop[2].extractShort();

    // Send a write command to the device
            DeviceData argin = new DeviceData();
            argin.insert("Hello World !");
            dev.command_inout("DevWriteMessage", argin);

    // Send a read command to the device
            DeviceData argout = dev.command_inout("DevReadMessage");
            String  received = argout.extractString();
            System.out.println(received);
        }
        catch (DevFailed e)
        {
           System.out.println(e);
        }
      }
    }
```

Modify this example to fit your device server or client's needs, compile it.

Do not forget when you start it to set the parameter TANGO_HOST with <host_name>:<port_number> (i.e. Serial -DTANGO_HOST=tango:20000 my_domain).

And forget about those painful early Tango days when you had to learn CORBA and manipulate Any's.

Life is going to easy and fun from now.

Part II
Data object classes

6 DeviceData class

This class manage data object for Tango device access.

6.1 Public methods

6.1.1 public DeviceData()

Constructor for DeviceData Object.

This method needs a database connection, that means that a *DevFailed* exception is thrown if the connection failed.

6.1.2 public void insert(<Tango type> argin)

Insert method for argin, where argin can be one of the Tango type (boolean, short, String[]...). This value will be used as argin parameter for the command_inout0 method.

6.1.3 public <TangoType> extract<Tango type>()

Extract the argout value of the command_inout() method.
i.e :
public short extractShort(): extract method for a short.
public short extractDouble(): extract method for a double.
public String extractString(): extract method for a String.
public String[] extractStringArray(): extract method for a String array.
public float[] extractFloatArray(): extract method for a float array.

6.2 Example

```
// Send a write command to the device
          DeviceData argin = new DeviceData();
          argin.insert("Hello World !");
          dev.command_inout("DevWriteMessage", argin);

// Send a read command to the device
          DeviceData argout = dev.command_inout("DevReadMessage");
          String  received = argout.extractString();
          System.out.println(received);
```

7 DbDatum

7.1 Public fields

7.1.1 public String name

The data name

7.2 public methods

7.2.1 public boolean is_empty()

This method does not throw exception.

- return true if the value has not been initialized.

7.2.2 public DbDatum(String name)

Constructor for DbDatum Object.

- parameter name The data name.

7.2.3 public DbDatum(String name, <Tango type> value)

Constructor for DbDatum Object.

- parameter name The data name.
- parameter value can be one of the Tango type (boolean, short, String[]...) and is the value to set the data.

7.2.4 public void insert(<Tango type> value)

Set the data value, where value can be one of the Tango type (boolean, short, String[]...).

7.2.5 public <TangoType> extract<Tango type>()

Extract the data value.
i.e :
public short extractShort(): extract method for a short.
public short extractDouble(): extract method for a double.
public String extractString(): extract method for a String.
public String[] extractStringArray(): extract method for a String array.
public float[] extractFloatArray(): extract method for a float array.

7.3 Example

```
// Update device properties.
          devname = "my/serial/device";
          DbDatum[] prop;
          prop = new DbDatum[3];
          prop[0] = new DbDatum("baudrate", 19200);
          prop[1] = new DbDatum("parity",   "none");
          prop[2] = new DbDatum("stopbits", 1);
          dbase.put_property(devname, prop);
```

8 DbDevInfo Class

Device information object.

8.1 Public fields

8.1.1 public String name

The device name.

8.1.2 public String _class

The class name.

8.1.3 public String server

The server name.

8.2 Public methods

8.2.1 public DbDevInfo()

Default constructor for DbDevInfo object.

8.2.2 public DbDevInfo(String name, String _class, String server)

Constructor for DbDevInfo object with values to set public fields.

8.3 Example

```
// Add a group of devices in the database
//--------------------------------------------
DbDevInfo[] devinfos;
devinfos = new DbDevInfo[2];
devinfos[0] = new DbDevInfo("sys/dummy/check3", "Dummy", "Dummy/check3");
devinfos[1] = new DbDevInfo("sys/dummy/check4", "Dummy", "Dummy/check3"); dbase.add_server(d
```

9 DbDevImportInfo class

This class is an object containing the imported device information.

9.1 Public fields

9.1.1 public String name

The device name.

9.1.2 public String ior

IOR connection as String.

9.1.3 public String version

TANGO protocol version number.

9.1.4 public boolean exported

true if device is exported.

9.2 Public methods

9.2.1 public DbDevImportInfo()

Default constructor.

9.3 Example

```
DbDevImportInfo  imp_info = dbase.import_device(devname); System.out.println(imp_info);
```

10 DbDevExportInfo class

This class is an object containing the exported device information.

10.1 Public fields

10.1.1 public String name

The device name.

10.1.2 public String ior

IOR connection as String.

10.1.3 public String host

Host name where device will be exported.

10.1.4 public String version

TANGO protocol version number.

10.1.5 public boolean exported

true if device is exported.

10.2 Public methods

10.2.1 public DbDevExportInfo()

Default constructor.

10.2.2 public DbDevExportInfo(String name, String ior, String host, String version)

Complete constructor.

10.3 Example

```
DbDevImportInfo imp_info = dbase.import_device(devname);

DbDevExportInfo exp_info =
    new DbDevExportInfo(devname, "MyServer/domain", imp_info.ior, "corvus", imp_info.version
dbase.export_device(exp_info);
```

Part III
Utility classes

11 ApiUtil class

This class manage a vector of Database object created. The goal of this class is to have ONLY ONE connection on a TANGO database for a host.

11.0.1 Database get_db_obj()

If no Databse object has been created before (no connection done) for the host specified in $TANGO_HOST, it create and return a new Database object. If a Database object has ben already created, it just returns this Datbase object.

11.0.2 Database get_db_obj(String hostname, String port)

If no Databse object has been created before (no connection done) for the host specified by hostname and port parameters, it create and return a new Database object. If a Database object has ben already created for this host, it just returns this Datbase object.

- parameter hostname : name of the host for TANGO database.
- parameter port : Port number (as String) for connection.

Part IV
Database access classes

12 Database class

12.1 General information methods

12.1.1 Creating a Database object

Do NOT use a Database constructor.

To manage an single connection an a host database, the Database object must be created through the ApiUtil class.

12.1.2 String get_info()

Query the database for general info about the table in the database.

- Return the result of the query as String.

```
Database  dbase = ApiUtil.get_db_obj();
String    info = dbase.get_info();
System.out.println(info);
```

12.1.3 String[] get_host_list()

Query the database for a list of host registred.

- Return the list of all hosts registred in TANGO database.

12.1.4 String[] get_host_list(String wildcard)

Query the database for a list of host registred.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the list of the hosts registred in TANGO database with the specified wildcard.

```
String[]  hosts = get_host_list("amber*");
for (int i=0 ; i<hosts.length ; i++)
    System.out.println(hosts[i]);
```

12.1.5 String[] get_server_list()

Query the database for a list of servers registred in the database.

- Return the list of all servers registred in TANGO database.

12.1.6 String[] get_server_list(String wildcard)

Query the database for a list of servers registred in the database.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the list of all servers registred in TANGO database.

12.1.7 String[] get_host_server_list(String hostname)

Query the database for a list of servers registred on the specified host.

- parameter hostname : the specified host name.
- Return the list of the servers registred in TANGO database for the specified host.

```
String[]  servers = get_host_server__list("corvus");
for (int i=0 ; i<servers.length ; i++)
    System.out.println(servers[i]);
```

12.1.8 void put_server_info(DbServInfo info)

Add/update server information in databse.

- parameter info : Server information for the specified server
- in a DbServinfo object.

```
DbServerInfo  info = new DbServerInfo("Serial/line1");
info.host = corvus;          // Will be registred on
info.controled = true;       // Will be controlled by Astor
info.startup_level = 4;      // Startup level used by Astor.
put_server_info(info);
```

12.1.9 DbServInfo get_server_info(String servname)

Query the database for server information.

- parameter servname : The specified server name.
- Return The information found for the specified server in a DBServInfo object.

12.2 Device Methods

12.2.1 void add_device(DbDevInfo devinfo)

Add/update a device to the database

- parameter devinfo : The device name, class and server specified in object.

```
DbDevInfo devinfo =
       new DbDevInfo("sys/database/1", "Dbase", "Databse/1");
dbase.add_device(devinfo);
```

12.2.2 void delete_device(String devname)

Delete the device of the specified name from the database

- parameter devname : The device name.

12.2.3 DbDevImportInfo import_device(String devname)

Query the database for the export info of the specified device.

- parameter devname : The device name.
- Return the information in a DbDevImportInfo.

```
DbDevImportInfo info = dbase.import_device(devname);
String name    = info.name;
String ior     = info.ior;      // IOR connection as String.
String version = info.version;  // TANGO protocol version number.
boolean exp    = info.exported; // true if device is exported.
```

12.2.4 void unexport_device(String servname)

Mark the specified server as unexported in the database.

- parameter sevrname : The server name.

12.2.5 void export_device(DbDevExportInfo devinfo)

Update the export info fort this device in the database.

- parameter devinfo : Device information to export.

```
DbDevExportInfo exp_info =
    new DbDevExportInfo(devname, "Serial/line1", imp_info.ior,
                        "corvus", imp_info.version);
dbase.export_device(exp_info);
```

12.2.6 String[] get_device_class_list(String servname)

Query the database for a list of devices and classes served by the specified server.

- parameter servname : The server name.
- Return the device names are stored in an array of strings.

12.2.7 String[] get_device_name(String servname, String classname)

Query the database for a list of devices served by the specified server and of the specified class.

- parameter servname : The server name.
- parameter clasname : The class name
- Return the device names are stored in an array of strings.

```
String[]  names = dbase.get_device_name("Serial/line1", "Serial");
```

12.2.8 String[] get_device_alias(String devname)

Query the database for a list of aliases for the specified device.

- parameter devvname : The server name.
- Return the device aliases are stored in an array of strings.

12.2.9 String[] get_device_domain(String wildcard)

Query the database for a list of device domain names witch match the wildcard provided.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the device domain are stored in an array of strings.

```
String[]   domains = get_device_domain("sy*");
```

12.2.10 String[] get_device_family(String wildcard)

Query the database for a list of device family names witch match the wildcard provided.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the device family are stored in an array of strings.

```
String[]   families = get_device_family("sys/*data*");
```

12.2.11 String[] get_device_member(String wildcard)

Query the database for a list of device member names witch match the wildcard provided.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the device member are stored in an array of strings.

```
String[]   members = get_device_member("sys/database/*");
```

12.3 Server methods

12.3.1 void add_server(String servname, DbDevInfo[] devinfos)

Add a group of devices to the database.

- parameter servname : Server name for these devices.
- parameter devinfo : Devices and server information.

```
DbDevInfo[]     devinfos;
devinfos = new DbDevInfo[2];
devinfos[0] = new DbDevInfo("sys/dummy/check3", "Dummy", "Dummy/check3");
devinfos[1] = new DbDevInfo("sys/dummy/check4", "Dummy", "Dummy/check3");
dbase.add_server(devinfos[0].name, devinfos);
```

12.3.2 void delete_server(String servname)

Delete the device server and its associated devices from the database.

- parameter servname : the server name.

12.3.3 void export_server(String devname, DbDevExportInfo[] devinfos)

Export a group of devices to the database. The device name, IOR, servr name etc are specified in the DbDevExportInfo array.

- parameter servname : server name for these devices.
- parameter devinfo : Devices and server information.

12.3.4 void unexport_server(String devname)

Mark all devices exported for this device server as unexported.

- parameter devname : the device name.

12.4 Object property methods

12.4.1 DbDatum[] get_property(String name, String[] propnames)

Query the database for a list of object (i.e. non-device) properties for the pecified object.

- parameter name : Object name.
- parameter propnames : list of property names.
- Retun properties in DbDatum objects.

```
String[]        propnames = { "Speed", "Temperatures" };
DbDatum[]       prop = dbase.get_property("my_object", data);
if (prop[0].is_empty()==false)
   System.out.println(prop[0].name + ": " + prop[0].extractDouble());
if (prop[1].is_empty()==false)
   System.out.println(prop[1].name + ": " + prop[1].extractFloat());
```

12.4.2 DbDatum get_property(String name, String propname)

Query the database for an object (i.e. non-device) property for the pecified object.

- parameter name : Object name.
- parameter propname : list of property names.
- Retun property in DbDatum object.

```
DbDatum  prop = dbase.get_property("my_object", “Speed”);

if (prop.is_empty()==false)
   System.out.println(prop.name + ": " + prop.extractDouble());
```

12.4.3 DbDatum[] get_property(String name, DbDatum[] properties)

Query the database for a list of object (i.e. non-device) properties for thr dpecified object. The property names are specified by the DbDatum array objects.

- parameter name : Object name.
- parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

```
DbDatum[]  datum;
datum = new DbDatum[2];
datum[0] = new DbDatum(“Speed”);
datum[1] = new DbDatum(“Temperature”);
DbDatum[]  prop = dbase.get_property("my_object", data);

if (prop[0].is_empty()==false)
   System.out.println(prop[0].name + ": " + prop[0].extractDouble());
if (prop[1].is_empty()==false)
   System.out.println(prop[1].name + ": " + prop[1].extractFloat());
```

12.4.4 void put_property(String name, DbDatum[] properties)

Insert or update a list of properties for the specified object The property names and their values are specified by the DbDatum array.

- parameter name : Object name.
- parameter properties : Properties names and values array.

```
DbDatum[]  datum;
datum = new DbDatum[2];
datum[0] = new DbDatum(“Speed”, 123.456);
datum[1] = new DbDatum(“Temperature”, 21.5);
dbase.put_property("my_object", data);
```

12.4.5 void delete_property(String name, String[] propnames)

Delete a list of properties for the specified object.

- parameter name : Object name.
- parameter propnames : Property names.

12.4.6 void delete_property(String name, String propname)

Delete a property for the specified object.

- parameter name : Object name.
- parameter propname : Property names.

12.4.7 void delete_property(String name, DbDatum[] properties)

Delete a list of properties for the specified object.

- parameter name : Object name.
- parameter properties : Property DbDatum objects.

12.5 Device property methods

12.5.1 String[] get_device_property_list(String devname, String wildcard)

Query the database for a list of device properties for the specified object.

- parameter devname : name of the specified device.
- parameter wildcard : filter (* matches any character e.g. a*).
- Return the property names in a String array.

12.5.2 DbDatum[] get_device_property(String name, String[] propnames)

Query the database for a list of device properties for the pecified object.

- parameter name : device name.
- parameter propnames : list of property names.
- Retun properties in DbDatum objects.

```
String[]        propnames = { "Speed", "Temperatures" };
DbDatum[]       prop = dbase.get_property("sys/motor/1", propnames);

if (prop[0].is_empty()==false)
   System.out.println(prop[0].name + ": " + prop[0].extractDouble());
if (prop[1].is_empty()==false)
   System.out.println(prop[1].name + ": " + prop[1].extractFloat());
```

12.5.3 DbDatum get_device_property(String name, String propname)

Query the database for a device property for the pecified object.

- parameter name : device name.
- parameter propname : property name.
- Retun property in DbDatum object.

```
DbDatum   prop = dbase.get_property("sys/motor/1", “Speed”);

if (prop.is_empty()==false)
   System.out.println(prop.name + ": " + prop.extractDouble());
```

12.5.4 DbDatum[] get_device_property(String name, DbDatum[] properties)

Query the database for a list of device properties for the pecified object. The property names are specified by the DbDatum array objects.

- parameter name : device name.
- parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

```
DbDatum[]  datum;
datum = new DbDatum[2];
datum[0] = new DbDatum("Speed");
datum[1] = new DbDatum("Temperature");
DbDatum[]  prop = dbase.get_property("sys/motor/1", datum);

if (prop[0].is_empty()==false)
   System.out.println(prop[0].name + ": " + prop[0].extractDouble());
if (prop[1].is_empty()==false)
   System.out.println(prop[1].name + ": " + prop[1].extractFloat());
```

12.5.5 void put_device_property(String name, DbDatum[] properties)

Insert or update a list of properties for the specified device The property names and their values are specified by the DbDatum array.

- parameter name : device name.
- parameter properties : Properties names and values array.

```
DbDatum[]  datum;
datum = new DbDatum[2];
datum[0] = new DbDatum("Speed", 123.456);
datum[1] = new DbDatum("Temperature", 21.5);
dbase.put_property("sys/motor/1", datum);
```

12.5.6 void delete_device_property(String name, String[] propnames)

Delete a list of properties for the specified object.

- parameter name : Device name.
- parameter propnames : Property names.

12.5.7 void delete_device_property(String name, String propname)

Delete a property for the specified object.

- parameter name : Device name.
- parameter propname : Property name.

12.5.8 void delete_device_property(String name, DbDatum[] properties)

Delete a list of properties for the specified object.

- parameter name : Device name.
- parameter properties : Property DbDatum objects.

12.5.9 DbDatum[] get_device_attribute_property(String name, String[] propnames)

Query the database for a list of device attribute properties for the specified object.

- parameter name : device name.
- parameter propnames : list of property names.
- Retun properties in DbDatum objects.

12.5.10 DbDatum get_device_attribute_property(String name, String propname)

Query the database for device attribute property for the specified object.

- parameter name : device name.
- parameter propname : property name.
- Retun property in DbDatum object.

12.5.11 DbDatum[] get_device_attribute_property(String name, DbDatum[] properties)

Query the database for a list of device attribute properties for the specified object. The property names are specified by the DbDatum array objects.

- parameter name : device name.
- parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

12.5.12 void put_device_attribute_property(String name, DbDatum[] properties)

Insert or update a list of properties for the specified device attribute. The property names and their values are specified by the DbDatum array.

- parameter name : device name.
- parameter properties : Properties names and values array.

12.5.13 void delete_device_attribute_property(String name, String[] propnames)

Delete a list of properties for the specified object.

- parameter name : Device name.
- parameter propnames : Property names.

12.5.14 void delete_device_attribute_property(String name, String propname)

Delete a property for the specified object.

- parameter name : Device name.
- parameter propname : Property name.

12.5.15 void delete_device_attribute_property(String name, DbDatum[] properties)

Delete a list of properties for the specified object.

- parameter name : Device name.
- parameter properties : Property DbDatum objects.

12.6 Class property methods

12.6.1 String[] get_class_property_list(String classname, String wildcard)

Query the database for a list of class properties for the specified object.

- parameter classname : name of the specified class.
- parameter wildcard : filter (* matches any character e.g. a*).
- Return the property names in a String array.

12.6.2 DbDatum[] get_class_property(String name, String[] propnames)

Query the database for a list of class properties for the pecified object.

- parameter name : Class name.
- parameter propnames : list of property names.
- Retun properties in DbDatum objects.

12.6.3 DbDatum get_class_property(String name, String propname)

Query the database for a class property for the pecified object.

- parameter name : Class name.
- parameter propname : list of property names.
- Retun property in DbDatum object.

12.6.4 DbDatum[] get_class_property(String name, DbDatum[] properties)

Query the database for a list of class properties for the pecified object. The property names are specified by the DbDatum array objects.

- parameter name : Class name.
- parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

12.6.5 void put_class_property(String name, DbDatum[] properties)

Insert or update a list of properties for the specified class. The property names and their values are specified by the DbDatum array.

- parameter name : Class name.
- parameter properties : Properties names and values array.

12.6.6 void delete_class_property(String name, String[] propnames)

Delete a list of properties for the specified object.

- parameter name : Class name.
- parameter propnames : Property names.

12.6.7 void delete_class_property(String name, String propname)

Delete a property for the specified object.

- parameter name : Class name.
- parameter propname : Property name.

12.6.8 void delete_class_property(String name, DbDatum[] properties)

Delete a list of properties for the specified object.

- parameter name : Class name.
- parameter properties : Property DbDatum objects.

12.6.9 DbDatum[] get_class_attribute_property(String name, String[] propnames)

Query the database for a list of class attribute properties for the pecified object.

- parameter name : Class name.
- parameter propnames : list of property names.
- Retun properties in DbDatum objects.

12.6.10 DbDatum get_class_attribute_property(String name, String propname)

Query the database for a class attribute property for the pecified object.

- parameter name : Class name.
- parameter propname : property name.
- Retun property in DbDatum object.

12.6.11 DbDatum[] get_class_attribute_property(String name, DbDatum[] properties)

Query the database for a list of class attribute properties for the pecified object. The property names are specified by the DbDatum array objects.

- parameter name : Class name.
- parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

12.6.12 void put_class_attribute_property(String name, DbDatum[] properties)

Insert or update a list of properties for the specified class attribute. The property names and their values are specified by the DbDatum array.

- parameter name : Class name.
- parameter properties : Properties names and values array.

12.6.13 void delete_class_attribute_property(String name, String[] propnames)

Delete a list of properties for the specified object.

- parameter name : Class name.
- parameter propnames : Property names.

12.6.14 void delete_class_attribute_property(String name, String propname)

Delete a property for the specified object.

- parameter name : Class name.
- parameter propname : Property names.

12.6.15 void delete_class_attribute_property(String name, DbDatum[] properties)

Delete a list of properties for the specified object.

- parameter name : Class name.
- parameter properties : Property DbDatum objects.

13 DbClass class

13.1 Class property methods

13.1.1 DbClass(String classname)

Database object used for TANGO database access.

13.1.2 String[] get_property_list(String wildcard)

Query the database for a list of class properties for this class.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the property names in a String array.

13.1.3 DbDatum[] get_property(String[] propnames)

Query the database for a list of properties for this class.

- parameter propnames : list of property names.
- Return properties in DbDatum objects.

13.1.4 DbDatum get_property(String propnames)

Query the database for a property for this class.

- parameter propname : property name.
- Return properties in DbDatum object.

13.1.5 DbDatum[] get_property(DbDatum[] properties)

Query the database for a list of properties for this class. The property names are specified by the DbDatum array objects.

- parameter properties : list of property DbDatum objects.
- Return properties in DbDatum objects.

13.1.6 void put_property(DbDatum[] properties)

Insert or update a list of properties for this class The property names and their values are specified by the DbDatum array.

- parameter properties : Properties names and values array.

13.1.7 void delete_property(String[] propnames)

Delete a list of properties for this class.

- parameter propnames : Property names.

13.1.8 void delete_property(String propname)

Delete a property for this class.

- parameter propname : Property name.

13.1.9 void delete_property(DbDatum[] properties)

Delete a list of properties for this class.

- parameter properties : Property DbDatum objects.

13.2 Class attribute property

13.2.1 void put_attibute_property(DbDatum[] properties)

Insert or update a list of properties for this class attribute. The property names and their values are specified by the DbDatum array.

- parameter properties : Properties names and values array.

13.2.2 void delete_attribute_property(String[] propnames)

Delete a list of properties for this object.

- parameter propnames : Property names.

13.2.3 delete_attribute_property(String propname)

Delete a property for this object.

- parameter propname : Property name.

13.2.4 delete_attribute_property(DbDatum[] properties)

Delete a list of properties for this object.

- parameter properties : Property DbDatum objects.

13.2.5 DbDatum[] get_attribute_property(String[] propnames)

Query the database for a list of class attribute properties for this device.

- parameter propnames : list of property names.
- Return properties in DbDatum objects.

13.2.6 DbDatum get_attribute_property(String propname)

Query the database for of class attribute property for this device.

- parameter propname : property name.
- Return property in DbDatum objects.

13.2.7 DbDatum[] get_attribute_property(DbDatum[] properties)

Query the database for a list of class attribute properties for this device. The property names are specified by the DbDatum array objects.

- parameter properties : list of property DbDatum objects.
- Return properties in DbDatum objects.

13.3 General information methods

13.3.1 String name()

This method does not throw any exception.

- return the class name.

14 DbServer class

This class manage database connection for Tango server.

14.0.2 public DbServer(String servname)

DbServer constructor.It makes a connection to the TANGO database for server management.

- Parameter servname : Name of the class oject.

14.0.3 public DbServer(String servname, Database dbase)

DbServer constructor.It makes a connection to the TANGO database for server management.

- Parameter servname : Name of the class oject.
- Parameter dbase : Database object previously created.

14.0.4 public DbServInfo get_info()

Query the database for server information.

- Return The information found for this server in a DbServInfo object.

```
DbServer   server = new DbServer("Serial/line1");
DbServInfo info = server.get_info();
System.out.println("Server name:   " + server.name);
System.out.println("Registred on:  " + server.host);
if (server.controled)
   System.out.println("Auto Start level:   " + server.startup_level);
```

14.0.5 public void put_info(DbServInfo info) throws DevFailed

Add/update server information in databse.

- Parameter info : Server information for this server in a DbServinfo object.

14.0.6 String[] get_device_class_list()

Query the database for a list of devices and classes served by this server.

- Return the device names are stored in an array of strings.

14.0.7 String[] get_device_name(String classname)

Query the database for a list of devices served by this server, for the specified class.

- parameter clasname : The class name
- Return the device names are stored in an array of strings.

```
String[]  names = server.get_device_name("Serial");
```

14.0.8 public String name()

This method does not throw any exception.

- return the server name.

Part V
Devices access classes

15 DeviceProxy class

15.1 Tango database management for Tango device.

The following methods are using database connection and could be used without exporting device.

15.1.1 public DbDevImportInfo import_device()

Query the database for the export info of this device.

- Return the information in a DbDevImportInfo.

15.1.2 public void export_device(DbDevExportInfo devinfo)

Update the export info for this device in the database.

- Parameter devinfo : Device information to export.

15.1.3 public void add_device(DbDevInfo devinfo)

Add/update this device to the database

- Parameter devinfo : The device name, class and server specified in object.

15.1.4 String[] get_property_list(String wildcard)

Query the database for a list of class properties for this device.

- parameter wildcard : filter (* matches any character e.g. a*).
- Return the property names in a String array.

15.1.5 public DbDatum[] get_property(String[] propnames)

Query the database for a list of device properties for this device.

- Parameter propnames : list of property names.
- Retun properties in DbDatum objects.

15.1.6 public DbDatum get_property(String propname)

Query the database for a device property for this device.

- Parameter propname : property name.
- Retun property in DbDatum objects.

```
// get device properties as from database.
    String[]          propnames = { "baudrate", "parity", "stopbits"};
    DeviceProxy       dev = new DeviceProxy("my/serial/device");
    prop = dev.get_property(propnames);
    if (prop[0].is_empty()==false)  baud   = prop[0].extractLong();
    if (prop[1].is_empty()==false)  parity = prop[1].extractString();
    if (prop[2].is_empty()==false)  stop   = prop[2].extractShort();
```

15.1.7 public DbDatum[] get_property(DbDatum[] properties)

Query the database for a list of device properties for this device. The property names are specified by the DbDatum array objects.

- Parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

15.1.8 public void put_property(DbDatum[] properties)

Insert or update a list of properties for this device. The property names and their values are specified by the DbDatum array.

- Parameter properties : Properties names and values array.

15.1.9 public void delete_property(String[] propnames)

Delete a list of properties for this device.

- Parameter propnames : Property names.

15.1.10 public void delete_property(String propname)

Delete a property for this device.
Parameter propname : Property name.

15.1.11 public void delete_property(DbDatum[] properties)

Delete a list of properties for this device.

- Parameter properties : Property DbDatum objects.

15.1.12 public void put_attribute_property(DbDatum[] properties)

Insert or update a list of properties for this device attribute. The property names and their values are specified by the DbDatum array.

- Parameter properties : Properties names and values array.

15.1.13 public void delete_attribute_property(String[] propnames)

Delete a list of properties for this object.

- Parameter propnames : Property names.

15.1.14 public void delete_attribute_property(String propname)

Delete a property for this object.

- Parameter propname : Property name.

15.1.15 public void delete_attribute_property(DbDatum[] properties)

Delete a list of properties for this object.

- Parameter properties : Property DbDatum objects.

15.1.16 public DbDatum[] get_attribute_property(String[] propnames)

Query the database for a list of device attribute properties for this device.

- Parameter propnames : list of property names.
- Retun properties in DbDatum objects.

15.1.17 public DbDatum get_attribute_property(String propname)

Query the database for a device attribute property for this device.

- Parameter propname : list of property name.
- Retun property in DbDatum object.

15.1.18 public DbDatum[] get_attribute_property(DbDatum[] properties)

Query the database for a list of device attribute properties for this device. The property names are specified by the DbDatum array objects.

- Parameter properties : list of property DbDatum objects.
- Retun properties in DbDatum objects.

15.1.19 public String name()

- Return the device name.

15.2 The exported device management methods

This class manage Tango device connection. It is an api between user and IDL Device object.

15.2.1 public DeviceProxy(String devname)

Constructor for a DeviceProxy object.

- Parameter devname : The device name.

15.2.2 public DeviceProxy(String devname, Database dbase)

DeviceProxy constructor. It will import the device.

- Parameter devname : name of the device to be imported.
- Parameter dbase : Database object created.

15.2.3 public Device set_timeout(int millis)

Change the timeout value for a device call.

- Parameter millis : New value of the timeout in milliseconds.

15.2.4 public int get_timeout()

- Return the value of the timeout in milliseconds.

15.2.5 public DeviceData command_inout(String command, DeviceData data)

Send a command to the device.

- Parameter command : Command name to send to the device.
- Parameter data : argin management object.

```
    DeviceProxy   dev = new DeviceProxy("my/serial/device");
// Send a write command to the device
    DeviceData argin = new DeviceData();
    argin.insert("Hello World !");
    dev.command_inout("DevWriteMessage", argin);

// Send a read command to the device
    DeviceData argout = dev.command_inout("DevReadMessage", data);
    String  received = argout.extractString();
    System.out.println(received);
```

15.2.6 public DeviceData command_inout(String command)

Send a command to the device.

- Parameter device : Device instance to send the command.
- Parameter command : Command name to send to the device.

```
    DeviceProxy  dev = new DeviceProxy("sr/powersupply/dipole");
// Send a DevOn command to the device
    dev.command_inout("DevOn");
```

15.2.7 public int ping()

Execute a ping command to the device.

- Return the elapsed time for the ping command in milliseconds.

15.2.8 public DevInfo info()

Execute an info command to the device.

15.2.9 public DevCmdInfo[] command_list_query()

Execute a command_list_query command to the device.