Reference Manual

1.0.0

Generated by Doxygen 1.5.3

Tue Dec 18 18:07:30 2007

Contents

Chapter 1

The Tango C Language Binding

The Tango C language binding is a reduced C interface which wraps the Tango C++ API. The actual binding only contains the basic query functionality and the basic synchronous reading and writing of commands and attributes.

The API is structured in proxy related functions, command related functions and attribute related functions.

Tango Proxy Related Functions (p. 17)

Tango Command Related Functions (p. 20)

Tango Attribute Related Functions (p. 23)

Tango Property and Database Related Functions (p. 28)

Error Handling Related Functions (p. 27)

Tango C Binding Enumerations (p. 7)

Tango Data Type Related Definitions (p. 11)

Tango C Binding Data Structures (p. 14)

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

Module Documentation

4.1 Tango C Binding Enumerations

4.1.1 Detailed Description

All enumerations used in the Tango C binding.

Enumerations

- enum **TangoDataType** {
  DEV_VOID = 0, **DEV_BOOLEAN**, **DEV_SHORT**, **DEV_LONG**,
  DEV_FLOAT, **DEV_DOUBLE**, **DEV_USHORT**, **DEV_ULONG**,
  DEV_STRING, **DEVVAR_CHARARRAY**, **DEVVAR_SHORTARRAY**, **DEVVAR_-LONGARRAY**,
  DEVVAR_FLOATARRAY, **DEVVAR_DOUBLEARRAY**, **DEVVAR_USHORTARRAY**, **DEVVAR_ULONGARRAY**,
  DEVVAR_STRINGARRAY, **DEVVAR_LONGSTRINGARRAY**, **DEVVAR_-DOUBLESTRINGARRAY**, **DEV_STATE**,
  CONST_DEV_STRING, **DEVVAR_BOOLEANARRAY**, **DEV_UCHAR**, **DEV_LONG64**,
  DEV_ULONG64, **DEVVAR_LONG64ARRAY**, **DEVVAR_ULONG64ARRAY**, **DEV_INT** }
- enum **TangoDevState** {
  ON, **OFF**, **CLOSE**, **OPEN**,
  INSERT, **EXTRACT**, **MOVING**, **STANDBY**,
  FAULT, **INIT**, **RUNNING**, **ALARM**,
  DISABLE, **UNKNOWN** }
- enum **AttrQuality** {
  ATTR_VALID, **ATTR_INVALID**, **ATTR_ALARM**, **ATTR_CHANGING**,
  ATTR_WARNING }
- enum **AttrWriteType** { **READ**, **READ_WITH_WRITE**, **WRITE**, **READ_WRITE** }
- enum **AttrDataFormat** { **SCALAR**, **SPECTRUM**, **IMAGE** }
- enum **DispLevel** { **OPERATOR**, **EXPERT** }
- enum **ErrSeverity** { **WARN**, **ERR**, **PANIC** }
- enum **DevSource** { **DEV**, **CACHE**, **CACHE_DEV** }

4.1.2 Enumeration Type Documentation

4.1.2.1 enum TangoDataType

All available Tango data types.

The list of all available Tango data types. Scalar types and array types.

Enumerator:

- ***DEV_VOID*** void
- ***DEV_BOOLEAN*** bool
- ***DEV_SHORT*** short
- ***DEV_LONG*** int (32bits)
- ***DEV_FLOAT*** float
- ***DEV_DOUBLE*** double
- ***DEV_USHORT*** unsigned short
- ***DEV_ULONG*** unsigned long
- ***DEV_STRING*** char ∗
- ***DEVVAR_CHARARRAY*** array of unsigned char
- ***DEVVAR_SHORTARRAY*** array of short
- ***DEVVAR_LONGARRAY*** array of int (32bits)
- ***DEVVAR_FLOATARRAY*** array of float
- ***DEVVAR_DOUBLEARRAY*** array of double
- ***DEVVAR_USHORTARRAY*** array of unsigned short
- ***DEVVAR_ULONGARRAY*** array of unsigned int (32bits)
- ***DEVVAR_STRINGARRAY*** array of char ∗
- ***DEVVAR_LONGSTRINGARRAY*** not yet supported in the C binding
- ***DEVVAR_DOUBLESTRINGARRAY*** not yet supported in the C binding
- ***DEV_STATE*** TangoDevState enumeration.
- ***CONST_DEV_STRING*** const char ∗
- ***DEVVAR_BOOLEANARRAY*** array of bool
- ***DEV_UCHAR*** unsigned char
- ***DEV_LONG64*** long or long long (64bits)
- ***DEV_ULONG64*** unsigned long or unsigned long long (64bits)
- ***DEVVAR_LONG64ARRAY*** array of long or long long (64bits)
- ***DEVVAR_ULONG64ARRAY*** array of unsigned long or unsigned long long (64bits)
- ***DEV_INT*** int (32bits)

4.1.2.2 enum TangoDevState

The Tango Device States.

The list of all possible states for Tango devices. Every state is reperesented with a fixed color on the application level.

Enumerator:

ON The device is switched ON (green).

OFF The device is switched OFF (white).

CLOSE The device is CLOSED (white).

OPEN The device is OPEN (green).

INSERT The device is INSERTED to the beam (white).

EXTRACT The device is EXTRACTED from the beam (green).

MOVING The device is MOVING or in a state transition (blue).

STANDBY The device is STANDBY (yellow).

FAULT The device has detected a FAULT (red).

INIT The device is INITialising (beige).

RUNNING The device is RUNNING and doing some work (blue).

ALARM The device indicates an ALARM (orange).

DISABLE The device is DISABLED by an interlock (magenta).

UNKNOWN The device lost its connection, the state is UNKNOWN (gray).

4.1.2.3 enum AttrQuality

The attribute quality factor.

The list of all possible attribute data quality factors. Every read attribute data has an assigned quality value to indicate the data validity.

Enumerator:

ATTR_VALID The attribute data is VALID.

ATTR_INVALID The attribute data is INVALID.

ATTR_ALARM The attribute indicates an ALARM on the data.

ATTR_CHANGING The attribute value is CHANGING and not stable.

ATTR_WARNING The attribute indicates an WARNING on the data.

4.1.2.4 enum AttrWriteType

The attribute write type.

The list of all possible attribute types. An attribute can be read only, write only or read/write.

Enumerator:

READ The attribute is read only.

READ_WITH_WRITE The attribute is of type read with a second write attribute associated.

WRITE The attribute is write only.

READ_WRITE The attribute is of type read/write.

4.1.2.5 enum AttrDataFormat

The attribute data format.

The data format of an attribute can be a scalar attribute, a spectrum (1D array) attribute or an image (2D array) attribute.

Enumerator:

SCALAR The attribute handles scalar values.

SPECTRUM The attribute handles a spectrum (1D array).

IMAGE The attribute handles an image (2D array).

4.1.2.6 enum DispLevel

The attribute display level.

The attribute might be displayed for expert users only.

Enumerator:

OPERATOR Attribute display all users.

EXPERT Attribute display only for expert users.

4.1.2.7 enum ErrSeverity

The error severity level.

A Tango error can indicate three different severity levels.

Enumerator:

WARN Warning level.

ERR Error level.

PANIC Real severe Panic level.

4.1.2.8 enum DevSource

The Tango data source.

Data can be read directly from the device or from the polling cache. In the case of CACHE_DEV, the data is read from the cache when it is available, otherwise from the device. This is the default setting.

Enumerator:

DEV Direct device reading.

CACHE Reading only from polling cache.

CACHE_DEV Reading from chache or device.

4.2 Tango Data Type Related Defi nitions

4.2.1 Detailed Description

Tango data type definitions for array data types and for long data types.

The long data types should be used to avoid the 32/64 bit problem.

Data Structures

- struct **VarBoolArray**

  A structure containing a pointer to a sequence of boolean values and the number of elements in the sequence.

- struct **VarCharArray**

  A structure containing a pointer to a sequence of char values and the number of elements in the sequence.

- struct **VarShortArray**

  A structure containing a pointer to a sequence of short values and the number of elements in the sequence.

- struct **VarUShortArray**

  A structure containing a pointer to a sequence of unsigned short values and the number of elements in the sequence.

- struct **VarLongArray**

  A structure containing a pointer to a sequence of 32 bit long values and the number of elements in the sequence.

- struct **VarULongArray**

  A structure containing a pointer to a sequence of 32 bit unsigned long values and the number of elements in the sequence.

- struct **VarLong64Array**

  A structure containing a pointer to a sequence of 64 bit long values and the number of elements in the sequence.

- struct **VarULong64Array**

  A structure containing a pointer to a sequence of 64 bit unsigned long values and the number of elements in the sequence.

- struct **VarFloatArray**

  A structure containing a pointer to a sequence of float values and the number of elements in the sequence.

- struct **VarDoubleArray**

  A structure containing a pointer to a sequence of double values and the number of elements in the sequence.

- struct **VarStringArray**

  A structure containing a pointer to a sequence of strings and the number of elements in the sequence.

- struct **VarStateArray**

  A structure containing a pointer to a sequence of TangoDevState values and the number of elements in the sequence.

- struct **VarBoolArray**

  A structure containing a pointer to a sequence of boolean values and the number of elements in the sequence.

- struct **VarCharArray**

  A structure containing a pointer to a sequence of char values and the number of elements in the sequence.

- struct **VarShortArray**

  A structure containing a pointer to a sequence of short values and the number of elements in the sequence.

- struct **VarUShortArray**

  A structure containing a pointer to a sequence of unsigned short values and the number of elements in the sequence.

- struct **VarLongArray**

  A structure containing a pointer to a sequence of 32 bit long values and the number of elements in the sequence.

- struct **VarULongArray**

  A structure containing a pointer to a sequence of 32 bit unsigned long values and the number of elements in the sequence.

- struct **VarLong64Array**

  A structure containing a pointer to a sequence of 64 bit long values and the number of elements in the sequence.

- struct **VarULong64Array**

  A structure containing a pointer to a sequence of 64 bit unsigned long values and the number of elements in the sequence.

- struct **VarFloatArray**

  A structure containing a pointer to a sequence of float values and the number of elements in the sequence.

- struct **VarDoubleArray**

  A structure containing a pointer to a sequence of double values and the number of elements in the sequence.

- struct **VarStringArray**

  A structure containing a pointer to a sequence of strings and the number of elements in the sequence.

- struct **VarStateArray**

  A structure containing a pointer to a sequence of TangoDevState values and the number of elements in the sequence.

Typedefs

- typedef int **TangoDevLong**

  A 32 bit long value.

- typedef unsigned int **TangoDevULong**

A 32 bit unsigned long value.

- typedef long long **TangoDevLong64**

  A 64 bit long value.

- typedef unsigned long long **TangoDevULong64**

  A 64 bit unsigned long value.

- typedef struct **VarBoolArray VarBoolArray**
- typedef struct **VarCharArray VarCharArray**
- typedef struct **VarShortArray VarShortArray**
- typedef struct **VarUShortArray VarUShortArray**
- typedef struct **VarLongArray VarLongArray**
- typedef struct **VarULongArray VarULongArray**
- typedef struct **VarLong64Array VarLong64Array**
- typedef struct **VarULong64Array VarULong64Array**
- typedef struct **VarFloatArray VarFloatArray**
- typedef struct **VarDoubleArray VarDoubleArray**
- typedef struct **VarStringArray VarStringArray**
- typedef struct **VarStateArray VarStateArray**

4.3 Tango C Binding Data Structures

4.3.1 Detailed Description

Data structures used in the Tango C binding.

Data Structures

- union **TangoAttributeData**

  An union of all Tango array data types used for attribute reading and writing.

- union **TangoCommandData**

  An union of all Tango scalar and array data types used for command data reading and writing.

- union **TangoPropertyData**

  An union of all Tango scalar and array data types used for property reading and writing.

- struct **CommandData**

  A structure containing the Tango data type and the command data union to transfer command data to and from a server.

- struct **AttributeData**

  A structure containing the scalar Tango data type and the attribute data union to transfer attribute data to and from a server.

- struct **AttributeDataList**

  A structure containing a pointer to a sequence of attribute data structures and the number of elements in the sequence.

- struct **DevFailed**

  A structure that maps all fields of the Tango::DevFailed exception.

- struct **ErrorStack**

  A structure containing a pointer to a sequence of error structures and the number of elements in the sequence.

- struct **CommandInfo**

  The command info structure contains descriptive command properties.

- struct **CommandInfoList**

  A structure containing a pointer to a sequence of command info structures and the number of elements in the sequence.

- struct **AttributeInfo**

  The attribute info structure contains descriptive attribute properties.

- struct **AttributeInfoList**

  A structure containing a pointer to a sequence of attribute info structures and the number of elements in the sequence.

- struct **DbDatum**

  A container structure for the Tango database access.

- struct **DbData**

  A structure containing a pointer to a sequence of ***DbDatum*** (p. 44) *structures and the number of elements in the sequence.*

- union **TangoAttributeData**

  An union of all Tango array data types used for attribute reading and writing.

- union **TangoCommandData**

  An union of all Tango scalar and array data types used for command data reading and writing.

- union **TangoPropertyData**

  An union of all Tango scalar and array data types used for property reading and writing.

- struct **CommandData**

  A structure containing the Tango data type and the command data union to transfer command data to and from a server.

- struct **AttributeData**

  A structure containing the scalar Tango data type and the attribute data union to transfer attribute data to and from a server.

- struct **AttributeDataList**

  A structure containing a pointer to a sequence of attribute data structures and the number of elements in the sequence.

- struct **DevFailed**

  A structure that maps all fields of the Tango::DevFailed exception.

- struct **ErrorStack**

  A structure containing a pointer to a sequence of error structures and the number of elements in the sequence.

- struct **CommandInfo**

  The command info structure contains descriptive command properties.

- struct **CommandInfoList**

  A structure containing a pointer to a sequence of command info structures and the number of elements in the sequence.

- struct **AttributeInfo**

  The attribute info structure contains descriptive attribute properties.

- struct **AttributeInfoList**

  A structure containing a pointer to a sequence of attribute info structures and the number of elements in the sequence.

- struct **DbDatum**

  A container structure for the Tango database access.

- struct **DbData**

  A structure containing a pointer to a sequence of ***DbDatum*** (p. 44) *structures and the number of elements in the sequence.*

Typedefs

- typedef union **TangoAttributeData TangoAttributeData**
- typedef union **TangoCommandData TangoCommandData**
- typedef union **TangoPropertyData TangoPropertyData**
- typedef struct **CommandData CommandData**
- typedef struct **AttributeData AttributeData**
- typedef struct **AttributeDataList AttributeDataList**
- typedef struct **DevFailed DevFailed**
- typedef struct **ErrorStack ErrorStack**
- typedef struct **CommandInfo CommandInfo**
- typedef struct **CommandInfoList CommandInfoList**
- typedef struct **AttributeInfo AttributeInfo**
- typedef struct **AttributeInfoList AttributeInfoList**
- typedef struct **DbDatum DbDatum**
- typedef struct **DbData DbData**

4.4 Tango Proxy Related Functions

4.4.1 Detailed Description

Functions to manipulate a device connection.

Functions

- bool **tango_create_device_proxy** (char ∗dev_name, void ∗∗proxy, **ErrorStack** ∗error)
  Create the access to a Tango device.
- bool **tango_delete_device_proxy** (void ∗∗proxy, **ErrorStack** ∗error)
  Delete the access to a Tango device.
- bool **tango_set_timeout_millis** (void ∗proxy, int millis, **ErrorStack** ∗error)
  Set the timeout of a device connection.
- bool **tango_get_timeout_millis** (void ∗proxy, int ∗millis, **ErrorStack** ∗error)
  Get the timeout of a device connection.
- bool **tango_set_source** (void ∗proxy, **DevSource** source, **ErrorStack** ∗error)
  Set the source for data reading.
- bool **tango_get_source** (void ∗proxy, **DevSource** ∗source, **ErrorStack** ∗error)
  Get the actual source for data reading.

4.4.2 Function Documentation

4.4.2.1 bool tango_create_device_proxy (char ∗ *dev_name*, void ∗∗ *proxy*, ErrorStack ∗ *error*)

Create the access to a Tango device.

Parameters:

← ***dev_name*** The name of the device to connect to.
→ ***proxy*** The pointer to the device handle.
→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.4.2.2 bool tango_delete_device_proxy (void ∗∗ *proxy*, ErrorStack ∗ *error*)

Delete the access to a Tango device.

Parameters:

← ***proxy*** The pointer to the device handle.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.4.2.3 bool tango_set_timeout_millis (void ∗ *proxy*, int *millis*, ErrorStack ∗ *error*)

Set the timeout of a device connection.

The timeout value is given in milliseconds.

Parameters:

← ***proxy*** The pointer to the device handle.
← ***millis*** The timout value.
→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.4.2.4 bool tango_get_timeout_millis (void ∗ *proxy*, int ∗ *millis*, ErrorStack ∗ *error*)

Get the timeout of a device connection.

The timeout value is given in milliseconds.

Parameters:

← ***proxy*** The pointer to the device handle.
→ ***millis*** The timout value.
→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.4.2.5 bool tango_set_source (void ∗ *proxy*, DevSource *source*, ErrorStack ∗ *error*)

Set the source for data reading.

Data can be read from the device or from the polling cache.

Parameters:

← ***proxy*** The pointer to the device handle.
← ***source*** The data source to use.
→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.4.2.6 bool tango_get_source (void $*$ *proxy*, DevSource $*$ *source*, ErrorStack $*$ *error*)

Get the actual source for data reading.

Data can be read from the device or from the polling cache.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\rightarrow$ ***source*** The actual data source.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.5 Tango Command Related Functions

4.5.1 Detailed Description

Functions to query and execute Tango commands.

Functions

- bool **tango_command_query** (void ∗proxy, char ∗cmd_name, **CommandInfo** ∗cmd_info, **ErrorStack** ∗error)

  Query the descriptive command properties for a given command.

- bool **tango_command_list_query** (void ∗proxy, **CommandInfoList** ∗cmd_info_list, **ErrorStack** ∗error)

  Query the descriptive command properties for all commands of a device.

- bool **tango_command_inout** (void ∗proxy, char ∗cmd_name, **CommandData** ∗argin, **CommandData** ∗argout, **ErrorStack** ∗error)

  Ececute Tango commands with input and output parameters.

- void **tango_free_CommandData** (**CommandData** ∗command_data)

  Free the allocated command output data.

- void **tango_free_CommandInfo** (**CommandInfo** ∗command_info)

  Free allocated command information.

- void **tango_free_CommandInfoList** (**CommandInfoList** ∗command_info_list)

  Free the list of all allocated command information structures.

4.5.2 Function Documentation

4.5.2.1 bool tango_command_query (void ∗ *proxy*, char ∗ *cmd_name*, CommandInfo ∗ *cmd_info*, ErrorStack ∗ *error*)

Query the descriptive command properties for a given command.

Parameters:

← ***proxy*** The pointer to the device handle.

← ***cmd_name*** The name of the command.

→ ***cmd_info*** Structure for descriptive command properties.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.5.2.2 bool tango_command_list_query (void $*$ *proxy*, CommandInfoList $*$ *cmd_info_list*, ErrorStack $*$ *error*)

Query the descriptive command properties for all commands of a device.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\rightarrow$ ***cmd_info_list*** A sequence of structures for descriptive command properties.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.5.2.3 bool tango_command_inout (void $*$ *proxy*, char $*$ *cmd_name*, CommandData $*$ *argin*, CommandData $*$ *argout*, ErrorStack $*$ *error*)

Ececute Tango commands with input and output parameters.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\leftarrow$ ***cmd_name*** The name of the command.

$\leftarrow$ ***argin*** The input parameters.

$\rightarrow$ ***argout*** The output parameters.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.5.2.4 void tango_free_CommandData (CommandData $*$ *command_data*)

Free the allocated command output data.

Parameters:

$\leftarrow$ ***command_data*** The command data structure with allocated fields.

4.5.2.5 void tango_free_CommandInfo (CommandInfo $*$ *command_info*)

Free allocated command information.

Parameters:

$\leftarrow$ ***command_info*** The command info structure with allocated fields.

4.5.2.6 void tango_free_CommandInfoList (CommandInfoList ∗ *command_info_list*)

Free the list of all allocated command information structures.

Parameters:

$\leftarrow$ ***command_info_list*** The sequence of command info structures with allocated fields.

4.6 Tango Attribute Related Functions

4.6.1 Detailed Description

Functions to query, read and write Tango attributes.

Reading of attribute properties is also possible.

Functions

- bool **tango_get_attribute_list** (void ∗proxy, **VarStringArray** ∗attr_names, **ErrorStack** ∗error)

  Get the names off all attributes of a device.

- bool **tango_get_attribute_config** (void ∗proxy, **VarStringArray** ∗attr_names, **AttributeInfoList** ∗attr_info_list, **ErrorStack** ∗error)

  Query the descriptive attribute properties for a list of attributes.

- bool **tango_attribute_list_query** (void ∗proxy, **AttributeInfoList** ∗attr_info_list, **ErrorStack** ∗error)

  Query the descriptive attribute properties for all attributes of a device.

- bool **tango_read_attribute** (void ∗proxy, char ∗attr_name, **AttributeData** ∗argout, **ErrorStack** ∗error)

  Read data from one attribute of a device.

- bool **tango_write_attribute** (void ∗proxy, **AttributeData** ∗argin, **ErrorStack** ∗error)

  Write data to one attribute of a device.

- bool **tango_read_attributes** (void ∗proxy, **VarStringArray** ∗attr_names, **AttributeDataList** ∗argout, **ErrorStack** ∗error)

  Read data from a list of attributes of a device.

- bool **tango_write_attributes** (void ∗proxy, **AttributeDataList** ∗argin, **ErrorStack** ∗error)

  Write data to a list of attributes of a device.

- void **tango_free_AttributeData** (**AttributeData** ∗attribute_data)

  Free the allocated attribute output data.

- void **tango_free_AttributeDataList** (**AttributeDataList** ∗attribute_data_list)

  Free the list of allocated attribute data structures.

- void **tango_free_VarStringArray** (**VarStringArray** ∗string_arr)

  Free the an allocated string array.

- void **tango_free_AttributeInfoList** (**AttributeInfoList** ∗attribute_info_list)

  Free the list of all allocated attribute information structures.

4.6.2 Function Documentation

4.6.2.1 bool tango_get_attribute_list (void ∗ *proxy*, VarStringArray ∗ *attr_names*, ErrorStack ∗ *error*)

Get the names off all attributes of a device.

Parameters:

← ***proxy*** The pointer to the device handle.

→ ***attr_names*** A string array with the attibute names.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.2 bool tango_get_attribute_config (void ∗ *proxy*, VarStringArray ∗ *attr_names*, AttributeInfoList ∗ *attr_info_list*, ErrorStack ∗ *error*)

Query the descriptive attribute properties for a list of attributes.

Parameters:

← ***proxy*** The pointer to the device handle.

← ***attr_names*** The string array with the attribute names.

→ ***attr_info_list*** A sequence of structures for descriptive attribute properties.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.3 bool tango_attribute_list_query (void ∗ *proxy*, AttributeInfoList ∗ *attr_info_list*, ErrorStack ∗ *error*)

Query the descriptive attribute properties for all attributes of a device.

Parameters:

← ***proxy*** The pointer to the device handle.

→ ***attr_info_list*** A sequence of structures for descriptive attribute properties.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.4 bool tango_read_attribute (void $*$ *proxy*, char $*$ *attr_name*, AttributeData $*$ *argout*, ErrorStack $*$ *error*)

Read data from one attribute of a device.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\leftarrow$ ***attr_name*** The attribute name.

$\rightarrow$ ***argout*** The read attribute data.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.5 bool tango_write_attribute (void $*$ *proxy*, AttributeData $*$ *argin*, ErrorStack $*$ *error*)

Write data to one attribute of a device.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\leftarrow$ ***argin*** The attribute data to be written.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.6 bool tango_read_attributes (void $*$ *proxy*, VarStringArray $*$ *attr_names*, AttributeDataList $*$ *argout*, ErrorStack $*$ *error*)

Read data from a list of attributes of a device.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\leftarrow$ ***attr_names*** The string array with the attribute names.

$\rightarrow$ ***argout*** A sequence of attribute data structures.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.7 bool tango_write_attributes (void $*$ *proxy*, AttributeDataList $*$ *argin*, ErrorStack $*$ *error*)

Write data to a list of attributes of a device.

Parameters:

$\leftarrow$ ***proxy*** The pointer to the device handle.

$\leftarrow$ ***argin*** A sequence of attribute data structures to be written.

$\rightarrow$ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.6.2.8 void tango_free_AttributeData (AttributeData $*$ *attribute_data*)

Free the allocated attribute output data.

Parameters:

$\leftarrow$ ***attribute_data*** The attribute data structure with allocated fields.

4.6.2.9 void tango_free_AttributeDataList (AttributeDataList $*$ *attribute_data_list*)

Free the list of allocated attribute data structures.

Parameters:

$\leftarrow$ ***attribute_data_list*** The sequence of attribute data structures with allocated fields.

4.6.2.10 void tango_free_VarStringArray (VarStringArray $*$ *string_arr*)

Free the an allocated string array.

Parameters:

$\leftarrow$ ***string_arr*** The allocated string array.

4.6.2.11 void tango_free_AttributeInfoList (AttributeInfoList $*$ *attribute_info_list*)

Free the list of all allocated attribute information structures.

Parameters:

$\leftarrow$ ***attribute_info_list*** The sequence of attribute info structures with allocated fields.

4.7 Error Handling Related Functions

4.7.1 Detailed Description

Functions to print and free a Tango error stack.

Functions

- void **tango_print_ErrorStack** (**ErrorStack** ∗error_stack)
  Print an error stack to stdout.
- void **tango_free_ErrorStack** (**ErrorStack** ∗error_stack)
  Free the data allocated for an error stack in case of a failure.

4.7.2 Function Documentation

4.7.2.1 void tango_print_ErrorStack (ErrorStack ∗ *error_stack*)

Print an error stack to stdout.

Parameters:

← ***error_stack*** The error stack of a Tango exception in case of failure.

4.7.2.2 void tango_free_ErrorStack (ErrorStack ∗ *error_stack*)

Free the data allocated for an error stack in case of a failure.

Parameters:

← ***error_stack*** The error stack of a Tango exception in case of failure.

4.8 Tango Property and Database Related Functions

4.8.1 Detailed Description

Functions to query, read and write Tango attributes.

Reading of attribute properties is also possible.

Functions

- bool **tango_create_database_proxy** (void ∗∗db_proxy, **ErrorStack** ∗error)

  Create the access to the Tango database.

- bool **tango_delete_database_proxy** (void ∗∗db_proxy, **ErrorStack** ∗error)

  Delete the access to the Tango database.

- bool **tango_get_device_exported** (void ∗db_proxy, char ∗name_filter, **DbDatum** ∗dev_list, **ErrorStack** ∗error)

  Get a list of exported devices using a name filter.

- bool **tango_get_device_exported_for_class** (void ∗db_proxy, char ∗class_name, **DbDatum** ∗dev_list, **ErrorStack** ∗error)

  Get a list of exported devices for a given Tango class.

- bool **tango_get_object_list** (void ∗db_proxy, char ∗name_filter, **DbDatum** ∗obj_list, **ErrorStack** ∗error)

  Get a list of free property objects from the Tango database using a name filter.

- bool **tango_get_object_property_list** (void ∗db_proxy, char ∗obj_name, char ∗name_filter, **DbDatum** ∗prop_list, **ErrorStack** ∗error)

  Get a list of property names for a given free property object, using a name filter.

- bool **tango_get_property** (void ∗db_proxy, char ∗obj_name, **DbData** ∗prop_list, **ErrorStack** ∗error)

  Get a list of properties for a given free property object.

- bool **tango_put_property** (void ∗db_proxy, char ∗obj_name, **DbData** ∗prop_list, **ErrorStack** ∗error)

  Put a list of properties for a given free property object.

- bool **tango_delete_property** (void ∗db_proxy, char ∗obj_name, **DbData** ∗prop_list, **ErrorStack** ∗error)

  Delete a list of properties for a given free property object.

- bool **tango_get_device_property** (void ∗dev_proxy, **DbData** ∗prop_list, **ErrorStack** ∗error)

  Get a list of device properties.

- bool **tango_put_device_property** (void ∗dev_proxy, **DbData** ∗prop_list, **ErrorStack** ∗error)

  Put a list of device properties.

- bool **tango_delete_device_property** (void ∗dev_proxy, **DbData** ∗prop_list, **ErrorStack** ∗error)
  Delete a list of device properties.
- void **tango_free_DbDatum** (**DbDatum** ∗db_datum)
  Free the allocated database data structure.
- void **tango_free_DbData** (**DbData** ∗db_data)
  Free the list of all allocated database data structures.

4.8.2 Function Documentation

4.8.2.1 bool tango_create_database_proxy (void ∗∗ *db_proxy*, ErrorStack ∗ *error*)

Create the access to the Tango database.

The function uses the environment variable “TANGO_HOST” to determine which instance of the TANGO database to connect to.

Parameters:

→ ***db_proxy*** The pointer to the database handle.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.2 bool tango_delete_database_proxy (void ∗∗ *db_proxy*, ErrorStack ∗ *error*)

Delete the access to the Tango database.

Parameters:

← ***db_proxy*** The pointer to the database handle.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.3 bool tango_get_device_exported (void ∗ *db_proxy*, char ∗ *name_filter*, DbDatum ∗ *dev_list*, ErrorStack ∗ *error*)

Get a list of exported devices using a name filter.

The name filter can contain one or more wilcards (∗). Example: sr/∗-pen/∗

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***name_filter*** The filter string

→ ***dev_list*** **DbDatum** (p. 44) structure containing a string array with the list of exported devices.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.4 bool tango_get_device_exported_for_class (void ∗ *db_proxy*, char ∗ *class_name*, DbDatum ∗ *dev_list*, ErrorStack ∗ *error*)

Get a list of exported devices for a given Tango class.

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***class_name*** The name of the Tango class.

→ ***dev_list*** **DbDatum** (p. 44) structure containing a string array with the list of exported devices.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.5 bool tango_get_object_list (void ∗ *db_proxy*, char ∗ *name_filter*, DbDatum ∗ *obj_list*, ErrorStack ∗ *error*)

Get a list of free property objects from the Tango database using a name filter.

The name filter can contain one or more wilcards (∗). Example: my∗prop/∗

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***name_filter*** The filter string

→ ***obj_list*** **DbDatum** (p. 44) structure containing a string array with the list of free property objects.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.6 bool tango_get_object_property_list (void ∗ *db_proxy*, char ∗ *obj_name*, char ∗ *name_filter*, DbDatum ∗ *prop_list*, ErrorStack ∗ *error*)

Get a list of property names for a given free property object, using a name filter.

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***obj_name*** The name of the free property object.

← ***name_filter*** The property name filter string

→ ***prop_list*** **DbDatum** (p. 44) structure containing a string array with the list of property names.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.7 bool tango_get_property (void ∗ *db_proxy*, char ∗ *obj_name*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Get a list of properties for a given free property object.

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***obj_name*** The name of the free property object.

→ ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the property names and the returned values.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.8 bool tango_put_property (void ∗ *db_proxy*, char ∗ *obj_name*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Put a list of properties for a given free property object.

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***obj_name*** The name of the free property object.

← ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the properties to write.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.9 bool tango_delete_property (void ∗ *db_proxy*, char ∗ *obj_name*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Delete a list of properties for a given free property object.

Parameters:

← ***db_proxy*** The pointer to the database handle.

← ***obj_name*** The name of the free property object.

← ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the property names to delete.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.10 bool tango_get_device_property (void ∗ *dev_proxy*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Get a list of device properties.

The function uses the device handle and not the database handle.

Parameters:

← ***dproxy*** The pointer to the device handle.

→ ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the property names and the returned values.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.11 bool tango_put_device_property (void ∗ *dev_proxy*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Put a list of device properties.

The function uses the device handle and not the database handle.

Parameters:

← ***dproxy*** The pointer to the device handle.

← ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the properties to write.

→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.12 bool tango_delete_device_property (void ∗ *dev_proxy*, DbData ∗ *prop_list*, ErrorStack ∗ *error*)

Delete a list of device properties.

The function uses the device handle and not the database handle.

Parameters:

← ***dproxy*** The pointer to the device handle.

← ***prop_list*** A sequence of **DbDatum** (p. 44) structures containing the property names to delete.
→ ***error*** The error stack of a Tango exception in case of failure.

Returns:

false when an failure was detected otherwise true.

4.8.2.13 void tango_free_DbDatum (DbDatum ∗ *db_datum*)

Free the allocated database data structure.

Parameters:

← ***db_datum*** The returned database data structure with allocated fields.

4.8.2.14 void tango_free_DbData (DbData ∗ *db_data*)

Free the list of all allocated database data structures.

Parameters:

← ***db_data*** The sequence of returned database data structures with allocated fields.

Chapter 5

Data Structure Documentation

5.1 AttributeData Struct Reference

5.1.1 Detailed Description

A structure containing the scalar Tango data type and the attribute data union to transfer attribute data to and from a server.

The structure also contains the data dimension, the data quality and a time stamp when the data was acquired.

Data Fields

- **TangoDataType data_type**
  Tango scalar data type.
- **TangoAttributeData attr_data**
  Union for attribute data.
- **AttrQuality quality**
  Data quality factor.
- char ∗ **name**
  Attribute name.
- int **dim_x**
  Data dimension X.
- int **dim_y**
  Data dimension Y.
- struct timeval **time_stamp**
  Time stanp in seconds and milliseconds since epoch.

5.2 AttributeDataList Struct Reference

5.2.1 Detailed Description

A structure containing a pointer to a sequence of attribute data structures and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **AttributeData** ∗ **sequence**

5.3 AttributeInfo Struct Reference

5.3.1 Detailed Description

The attribute info structure contains descriptive attribute properties.

Data Fields

- char ∗ **name**

  Attribute name string.

- **AttrWriteType writable**

  Attribute type READ, WRITE, READ and WRITE.

- **AttrDataFormat data_format**

  scalar, 1D or 2D data

- **TangoDataType data_type**

  The scalar Tango data type.

- int **max_dim_x**

  Maximum data size X.

- int **max_dim_y**

  Maximum data size Y.

- char ∗ **description**

  Attribute description text.

- char ∗ **label**

  Attribute GUI label.

- char ∗ **unit**

  Attribute unit.

- char ∗ **standard_unit**

  Conversion factor to MKS unit.

- char ∗ **display_unit**

  Conversion factor to GUI display unit.

- char ∗ **format**

  Attribute diplay format (printf format).

- char ∗ **min_value**

  Min value, checked when writing to an attribute.

- char ∗ **max_value**

  Max value, checked when writing to an attribute.

- char ∗ **min_alarm**

  Min alarm value, checked during state reading.

- char ∗ **max_alarm**

  Max alarm value, checked during state reading.

- char ∗ **writable_attr_name**

  Used only for READ_WTH_WRITE attributes.

- **DispLevel disp_level**

  operator or expert display

5.4 AttributeInfoList Struct Reference

5.4.1 Detailed Description

A structure containing a pointer to a sequence of attribute info structures and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **AttributeInfo** $*$ **sequence**

5.5 CommandData Struct Reference

5.5.1 Detailed Description

A structure containing the Tango data type and the command data union to transfer command data to and from a server.

Data Fields

- **TangoDataType arg_type**

  Tango data type.

- **TangoCommandData cmd_data**

  Union for command data.

5.6 CommandInfo Struct Reference

5.6.1 Detailed Description

The command info structure contains descriptive command properties.

Data Fields

- char ∗ **cmd_name**

  Command name string.

- int **cmd_tag**

  Command as binary value (for TACO).

- int **in_type**

  in type as binary value

- int **out_type**

  out type as binary value

- char ∗ **in_type_desc**

  description of in type (optional)

- char ∗ **out_type_desc**

  description of out type (optional)

- **DispLevel disp_level**

  Command display level.

5.7 CommandInfoList Struct Reference

5.7.1 Detailed Description

A structure containing a pointer to a sequence of command info structures and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **CommandInfo** ∗ **sequence**

5.8 DbData Struct Reference

5.8.1 Detailed Description

A structure containing a pointer to a sequence of **DbDatum** (p. 44) structures and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **DbDatum** ∗ **sequence**

5.9 DbDatum Struct Reference

5.9.1 Detailed Description

A container structure for the Tango database access.

All Tango query data and property releated data is passed with this structure.

Data Fields

- char ∗ **property_name**

  Name of the property.

- **TangoDataType data_type**

  Tango data type.

- **TangoPropertyData prop_data**

  Union for property data.

- bool **is_empty**

  set when no properties available

- bool **wrong_data_type**

  set when the property value cannot be converted to the given data type

5.10 DevFailed Struct Reference

5.10.1 Detailed Description

A structure that maps all fields of the Tango::DevFailed exception.

Data Fields

- char ∗ **desc**

  Error description.

- char ∗ **reason**

  Error reason.

- char ∗ **origin**

  Error origin (class and method).

- **ErrSeverity severity**

  Error severity.

5.11 ErrorStack Struct Reference

5.11.1 Detailed Description

A structure containing a pointer to a sequence of error structures and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **DevFailed** ∗ **sequence**

5.12 TangoAttributeData Union Reference

5.12.1 Detailed Description

An union of all Tango array data types used for attribute reading and writing.

Data Fields

- **VarBoolArray bool_arr**
- **VarCharArray char_arr**
- **VarShortArray short_arr**
- **VarUShortArray ushort_arr**
- **VarLongArray long_arr**
- **VarULongArray ulong_arr**
- **VarLong64Array long64_arr**
- **VarULong64Array ulong64_arr**
- **VarFloatArray float_arr**
- **VarDoubleArray double_arr**
- **VarStringArray string_arr**
- **VarStateArray state_arr**

5.13 TangoCommandData Union Reference

5.13.1 Detailed Description

An union of all Tango scalar and array data types used for command data reading and writing.

Data Fields

- bool **bool_val**
- short **short_val**
- unsigned short **ushort_val**
- int **long_val**
- unsigned int **ulong_val**
- float **float_val**
- double **double_val**
- char ∗ **string_val**
- **TangoDevState state_val**
- **TangoDevLong64 long64_val**
- **TangoDevULong64 ulong64_val**
- **VarBoolArray bool_arr**
- **VarCharArray char_arr**
- **VarShortArray short_arr**
- **VarUShortArray ushort_arr**
- **VarLongArray long_arr**
- **VarULongArray ulong_arr**
- **VarLong64Array long64_arr**
- **VarULong64Array ulong64_arr**
- **VarFloatArray float_arr**
- **VarDoubleArray double_arr**
- **VarStringArray string_arr**
- **VarStateArray state_arr**

5.14 TangoPropertyData Union Reference

5.14.1 Detailed Description

An union of all Tango scalar and array data types used for property reading and writing.

Data Fields

- bool **bool_val**
- unsigned char **char_val**
- short **short_val**
- unsigned short **ushort_val**
- int **long_val**
- unsigned int **ulong_val**
- float **float_val**
- double **double_val**
- char ∗ **string_val**
- **TangoDevLong64 long64_val**
- **TangoDevULong64 ulong64_val**
- **VarShortArray short_arr**
- **VarUShortArray ushort_arr**
- **VarLongArray long_arr**
- **VarULongArray ulong_arr**
- **VarLong64Array long64_arr**
- **VarULong64Array ulong64_arr**
- **VarFloatArray float_arr**
- **VarDoubleArray double_arr**
- **VarStringArray string_arr**

5.15 VarBoolArray Struct Reference

5.15.1 Detailed Description

A structure containing a pointer to a sequence of boolean values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- bool ∗ **sequence**

5.16 VarCharArray Struct Reference

5.16.1 Detailed Description

A structure containing a pointer to a sequence of char values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- unsigned char ∗ **sequence**

5.17 VarDoubleArray Struct Reference

5.17.1 Detailed Description

A structure containing a pointer to a sequence of double values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- double $*$ **sequence**

5.18 VarFloatArray Struct Reference

5.18.1 Detailed Description

A structure containing a pointer to a sequence of float values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- float ∗ **sequence**

5.19 VarLong64Array Struct Reference

5.19.1 Detailed Description

A structure containing a pointer to a sequence of 64 bit long values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **TangoDevLong64** $*$ **sequence**

5.20 VarLongArray Struct Reference

5.20.1 Detailed Description

A structure containing a pointer to a sequence of 32 bit long values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **TangoDevLong** ∗ **sequence**

5.21 VarShortArray Struct Reference

5.21.1 Detailed Description

A structure containing a pointer to a sequence of short values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- short ∗ **sequence**

5.22 VarStateArray Struct Reference

5.22.1 Detailed Description

A structure containing a pointer to a sequence of TangoDevState values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **TangoDevState** ∗ **sequence**

5.23 VarStringArray Struct Reference

5.23.1 Detailed Description

A structure containing a pointer to a sequence of strings and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- char ** **sequence**

5.24 VarULong64Array Struct Reference

5.24.1 Detailed Description

A structure containing a pointer to a sequence of 64 bit unsigned long values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **TangoDevULong64** $*$ **sequence**

5.25 VarULongArray Struct Reference

5.25.1 Detailed Description

A structure containing a pointer to a sequence of 32 bit unsigned long values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- **TangoDevULong** $*$ **sequence**

5.26 VarUShortArray Struct Reference

5.26.1 Detailed Description

A structure containing a pointer to a sequence of unsigned short values and the number of elements in the sequence.

Data Fields

- unsigned int **length**
- unsigned short * **sequence**

Index