
TMC Prototype Documentation

Release 1.0

NCRA India

Jan 08, 2021

Contents:

1	Central Node	1
2	Central Node	7
3	Subarray Node	17
4	Subarray Node	27
5	Dish Leaf Node	33
6	Dish Master	41
7	CSP Master Leaf Node	43
8	SDP Subarray Leaf Node	49
9	Subarray Node	51
10	CSP Subarray Leaf Node	73
11	SDP Master Leaf Node	75
12	MCCS Master Leaf Node	83
13	MCCS Subarray Leaf Node	89
14	Indices and tables	95
	Python Module Index	97
	Index	99

CHAPTER 1

Central Node

Central Node is a coordinator of the complete M&C system. Central Node implements the standard set of state and mode attributes defined by the SKA Control Model.

```
class tmcprototype.centralnode.src.centralnode.central_node.CentralNode(*args,  
**kwargs)
```

Central Node is a coordinator of the complete M&C system.

AssignResources (argin)

AssignResources command invokes the AssignResources command on lower level devices.

CentralAlarmHandler

Used by autodoc_mock_imports.

CspMasterLeafNodeFQDN

Used by autodoc_mock_imports.

DishLeafNodePrefix

Used by autodoc_mock_imports.

```
class InitCommand(*args, **kwargs)
```

A class for the TMC CentralNode's init_device() method.

do ()

Initializes the attributes and properties of the Central Node.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if error occurs while initializing the CentralNode device or if error occurs while creating device proxy for any of the devices like SubarrayNode, DishLeafNode, CSPMasterLeafNode or SDPMMasterLeafNode.

NumDishes

Used by autodoc_mock_imports.

ReleaseResources (argin)

Release all the resources assigned to the given Subarray.

SdpMasterLeafNodeFQDN

Used by autodoc_mock_imports.

StandByTelescope ()

This command invokes SetStandbyLPMode() command on DishLeafNode, StandBy() command on Csp-MasterLeafNode and SdpMasterLeafNode and Off() command on SubarrayNode and sets CentralNode into OFF state.

StartUpTelescope ()

This command invokes SetOperateMode() command on DishLeadNode, On() command on Csp-MasterLeafNode, SdpMasterLeafNode and SubarrayNode and sets the Central Node into ON state.

StowAntennas (argin)

This command stows the specified receptors.

TMAlarmHandler

Used by autodoc_mock_imports.

TMMidSubarrayNodes

Used by autodoc_mock_imports.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook ()

Internal construct of TANGO.

delete_device ()

Internal construct of TANGO.

init_command_objects ()

Initialises the command handlers for commands supported by this device.

is_AssignResources_allowed ()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_ReleaseResources_allowed ()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_StandByTelescope_allowed ()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

is_StartUpTelescope_allowed ()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

is_StowAntennas_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

read_activityMessage()

Internal construct of TANGO. Returns activity message.

read_subarray1HealthState()

Internal construct of TANGO. Returns Subarray1 health state.

read_subarray2HealthState()

Internal construct of TANGO. Returns Subarray2 health state.

read_subarray3HealthState()

Internal construct of TANGO. Returns Subarray3 health state.

read_telescopeHealthState()

Internal construct of TANGO. Returns the Telescope health state.

subarray1HealthState

Used by autodoc_mock_imports.

subarray2HealthState

Used by autodoc_mock_imports.

subarray3HealthState

Used by autodoc_mock_imports.

telescopeHealthState

Used by autodoc_mock_imports.

write_activityMessage(value)

Internal construct of TANGO. Sets the activity message.

```
tmcprototype.centralnode.src.centralnode.central_node.main(args=None,
                                                       **kwargs)
```

Runs the CentralNode. :param args: Arguments internal to TANGO

Parameters **kwargs** – Arguments internal to TANGO

Returns CentralNode TANGO object.

```
tmcprototype.centralnode.src.centralnode.central_node.AssignResources
```

Used by autodoc_mock_imports.

```
tmcprototype.centralnode.src.centralnode.central_node.DeviceData
```

Used by autodoc_mock_imports.

```
tmcprototype.centralnode.src.centralnode.central_node.ObsStateAggregator
```

Used by autodoc_mock_imports.

```
tmcprototype.centralnode.src.centralnode.central_node.ReleaseResources
```

Used by autodoc_mock_imports.

```
tmcprototype.centralnode.src.centralnode.central_node.StandByTelescope
```

Used by autodoc_mock_imports.

```
tmcprototype.centralnode.src.centralnode.central_node.StartUpTelescope
```

Used by autodoc_mock_imports.

tmcprototype.centralnode.src.centralnode.central_node.**StowAntennas**

Used by autodoc_mock_imports.

AssignResources class for CentralNode.

class tmcprototype.centralnode.src.centralnode.assign_resources_command.**AssignResources** (*ar*

**k

A class for CentralNode's AssignResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Assigns resources to given subarray. It accepts the subarray id, receptor id list and SDP block in JSON string format. Upon successful execution, the 'receptorIDList' attribute of the given subarray is populated with the given receptors. Also checking for duplicate allocation of resources is done. If already allocated it will throw error message regarding the prior existence of resource.

Parameters **argin** – The string in JSON format. The JSON contains following values:

subarrayID: DevShort. Mandatory.

dish: Mandatory JSON object consisting of

receptorIDList: DevVarStringArray The individual string should contain dish numbers in string format with preceding zeroes upto 3 digits. E.g. 0001, 0002.

sdp: Mandatory JSON object consisting of

id: DevString The SBI id.

max_length: DevDouble Maximum length of the SBI in seconds.

scan_types: array of the blocks each consisting following parameters id:

DevString The scan id.

coordinate_system: DevString

ra: DevString

Dec: DevString

processing_blocks: array of the blocks each consisting following parameters id:

DevString The Processing Block id.

workflow:

type: DevString

id: DevString

version: DevString

parameters: {}

Example: {"subarrayID":1,"dish":{"receptorIDList":["0001","0002"]},"sdp":{"id":"sbi-mvp01-20200325-00001","max_length":100.0,"scan_types":[{"id":"science_A","coordinate_system":"ICRS","ra":"02:42:45.000","dec":"-00:00:47.84","channels":[{"count":744,"start":0,"stride":2,"freq_min":100.0,"freq_max":150.0}]}]}

```

0.35e9,"freq_max":0.368e9,"link_map":[[0,0],[200,1],[744,2],[944,3]]},{“count”:744,”start”:2000,
“stride”:1,”freq_min”:0.36e9,”freq_max”:0.368e9,”link_map”:[[2000,4],[2200,5]]]},{{“id”:
“calibration_B”,”coordinate_system”：“ICRS”,”ra”：“12:29:06.699”,”dec”：“02:03:08.598”,”chan-
nels”:[{“count”:744,”start”:0,”stride”:2,”freq_min”:0.35e9,”freq_max”:0.368e9,”link_map”:
[[0,0],[200,1],[744,2],[944,3]]},{“count”:744,”start”:2000,”stride”:1,”freq_min”:0.36e9,
“freq_max”:0.368e9,”link_map”:[[2000,4],[2200,5]]}]},”processing_blocks”:[{{“id”：“pb-
mvp01-20200325-00001”,”workflow”:{“type”：“realtime”,”id”：“vis_receive”,”version”:
“0.1.0”},”parameters”:{}},{{“id”：“pb-mvp01-20200325-00002”,”workflow”:{“type”：“realtime”,
“id”：“test_realtime”,”version”：“0.1.0”},”parameters”:{}},{{“id”：“pb-mvp01-20200325-00003”,
“workflow”:{“type”：“batch”,”id”：“ical”,”version”：“0.1.0”},”parameters”:{}},”dependencies”:
[{{“pb_id”：“pb-mvp01-20200325-00001”,”type”:[“visibilities”]}},{{“id”：“pb-mvp01-20200325-
00004”,”workflow”:{“type”：“batch”,”id”：“dpreb”,”version”：“0.1.0”},”parameters”:{}},”dependencies”:
[{{“pb_id”：“pb-mvp01-20200325-00003”,”type”:[“calibration”]}]}]}]}

```

Note: From Jive, enter above input string without any space.

Returns

A tuple containing a return code and a string in JSON format on successful assignment of given resources. The JSON string contains following values:

dish: Mandatory JSON object consisting of

receptorIDList_success: DevVarStringArray Contains ids of the receptors which are successfully allocated. Empty on unsuccessful allocation.

Example: { “dish”: { “receptorIDList_success”: [“0001”, “0002”] } }

Return type

None

Raises DevFailed when the API fails to allocate resources.

Note: Enter input without spaces as: {“dish”: {“receptorIDList_success”: [“0001”, “0002”] } }

ReleaseResources class for CentralNode.

```
class tmcprototype.centralnode.src.centralnode.release_resources_command.ReleaseResources(
```

A class for CentralNode’s ReleaseResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Release all the resources assigned to the given Subarray. It accepts the subarray id, releaseALL flag and receptorIDList in JSON string format. When the releaseALL flag is True, ReleaseAllResources command is invoked on the respective SubarrayNode. In this case, the receptorIDList tag is empty as all the resources of the Subarray are to be released. When releaseALL is False, ReleaseResources will be invoked on the SubarrayNode and the resources provided in receptorIDList tag, are to be released from the Subarray. The selective release of the resources when releaseALL Flag is False is not yet supported.

Parameters argin – The string in JSON format. The JSON contains following values:

subarrayID: DevShort. Mandatory.

releaseALL: Boolean(True or False). Mandatory. True when all the resources to be released from Subarray.

receptorIDList: DevVarStringArray. Empty when releaseALL tag is True.

Example:

```
{ "subarrayID": 1, "releaseALL": true, "receptorIDList": [] }
```

Note: From Jive, enter input as: {"subarrayID":1,"releaseALL":true,"receptorIDList":[]}
without any space.

Returns

A tuple containing a return code and a string in json format on successful release of all the resources. The JSON string contains following values:

releaseALL: Boolean(True or False). If True, all the resources are successfully released from the Subarray.

receptorIDList: DevVarStringArray. If releaseALL is True, receptorIDList is empty. Else list returns resources (device names) that are noe released from the subarray.

Example: argout = {

```
"ReleaseAll" : True, "receptorIDList" : [] }
```

rtype None

raises ValueError if input argument json string contains invalid value KeyError if input argument json string contains invalid key DevFailed if the command execution or command invocation on SubarrayNode is not successful

StowAntennas class for CentralNode.

```
class tmcpototype.centralnode.src.centralnode.stow_antennas_command.StowAntennas(*args, **kwargs)
```

A class for CentralNode's StowAntennas() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Invokes the command SetStowMode on the specified receptors.

Parameters **argin** – List of Receptors to be stowed.

Returns None

Raises DevFailed if error occurs while invoking command of DishLeafNode ValueError if error occurs if input argument json string contains invalid value

CHAPTER 2

Central Node

Central Node is a coordinator of the complete M&C system. Central Node implements the standard set of state and mode attributes defined by the SKA Control Model.

```
class tmcprototype.centralnode_low.src.centralnode_low.central_node_low.CentralNode (*args,  
**kwargs)
```

Central Node is a coordinator of the complete M&C system.

AssignResources (*argin*)

AssignResources command invokes the AssignResources command on lower level devices.

CentralAlarmHandler

Used by autodoc_mock_imports.

```
class InitCommand (*args, **kwargs)
```

A class for the TMC CentralNode's init_device() method.

do ()

Initializes the attributes and properties of the Central Node Low.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if error occurs while initializing the CentralNode device or if error occurs while creating device proxy for any of the devices like SubarrayNodeLow or MccsMasterLeafNode.

MCCSMasterLeafNodeFQDN

Used by autodoc_mock_imports.

ReleaseResources (*argin*)

Release all the resources assigned to the given Subarray.

StandByTelescope ()

This command invokes Off() command on SubarrayNode, MCCSMasterLeafNode and sets CentralNode into OFF state.

StartUpTelescope()

This command invokes On() command on SubarrayNode, MCCSMasterLeafNode and sets the Central Node into ON state.

TMAlarmHandler

Used by autodoc_mock_imports.

TMLowSubarrayNodes

Used by autodoc_mock_imports.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

delete_device()

Internal construct of TANGO.

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_AssignResources_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state

Return type boolean

is_ReleaseResources_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_StandByTelescope_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_StartUpTelescope_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

read_activityMessage()

Internal construct of TANGO. Returns activity message.

read_subarray1HealthState()

Internal construct of TANGO. Returns Subarray1 health state.

read_telescopeHealthState()

Internal construct of TANGO. Returns the Telescope health state.

subarray1HealthState

Used by autodoc_mock_imports.

telescopeHealthState

Used by autodoc_mock_imports.

write_activityMessage (value)

Internal construct of TANGO. Sets the activity message.

```
tmcprototype.centralnodelow.src.centralnodelow.central_node_low.main (args=None,
**kwargs)
```

Runs the CentralNode. :param args: Arguments internal to TANGO

Parameters **kwargs** – Arguments internal to TANGO

Returns CentralNode TANGO object.

```
class tmcprototype.centralnodelow.src.centralnodelow.central_node_low.AssignResources (*args,
**kwargs)
```

A class for CentralNode's AssignResources() command.

assign_resources_leaf_node (tango_client, input_arg)

Invokes assign Resources command on leaf node with input argument.

Parameters

- **tango_client** – client of corresponding leaf node
- **input_arg** – Json string input to invoke command.

Raises DevFailed if error occurs while invoking command on any of the devices like SubarrayNode, MCCSMasterLeafNode

assign_resources_mccs (json_argument, device_data)

Create proxy of mccs master leaf node and call method assign_resources_leaf_node.

Parameters **json_argument** – The string in JSON format. **device_data** : Object of class device_data.

return: None

assign_resources_subarray (json_argument, device_data)

Delete subarray id from json argument and create proxy of subarray corresponding to subarray id and call assign_resources_leaf_node method.

Parameters **json_argument** – The string in JSON format. **device_data** : Object of class device_data

Returns None

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Assigns resources to given subarray. It accepts the subarray id, station ids, station beam id and channels in JSON string format.

Parameters **argin** – The string in JSON format. The JSON contains following values:

subarray_id: DevShort. Mandatory. Sub-Array to allocate resources to

station_ids: DevArray. Mandatory list of stations contributing beams to the data set

channels: DevArray. Mandatory list of frequency channels used

station_beam_ids: DevArray. Mandatory logical ID of beam

Example:

```
{ "subarray_id": 1, "station_ids": [1,2], "channels": [1,2,3,4,5,6,7,8], "station_beam_ids": [1]
```

```
}
```

Note: Enter input without spaces as:{“subarray_id”:1,”station_ids”:[1,2],”channels”:[1,2,3,4,5,6,7,8],”station_beam_ids”:[1]}

Returns None

Raises KeyError if input argument json string contains invalid key ValueError if input argument json string contains invalid value

```
class tmcprototype.centralnode_low.src.centralnode_low.central_node_low.ReleaseResources(*args, **kwargs)
```

A class for CentralNodeLow's ReleaseResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Release all the resources assigned to the given Subarray. It accepts the subarray id, release_all flag in JSON string format. When the release_all flag is True, ReleaseAllResources command is invoked on the respective SubarrayNode.

Parameters **argin** – The string in JSON format. The JSON contains following values:

subarray_id: DevShort. Mandatory.

release_all: Boolean(True or False). Mandatory. True when all the resources to be released from Subarray.

Example:

```
{ "subarray_id": 1, "release_all": true,
```

```
}
```

Note: From Jive, enter input as: {“subarray_id”:1,”release_all”:true} without any space.

Raises ValueError if input argument json string contains invalid value KeyError if input argument json string contains invalid key DevFailed if the command execution or command invocation on SubarrayNode is not successful

```
class tmcprototype.centralnode_low.src.centralnode_low.central_node_low.StandByTelescope(*args, **kwargs)
```

A class for Low CentralNode's StandByTelescope() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do ()

Sets the CentralNodeLow into OFF state. Invokes the respective command on lower level nodes and devices.

param argin: None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (resultCode, str)

standby_leaf_node (*tango_client*)

Invoke command Off leaf nodes.

Parameters **tango_client** – proxy of corresponding leaf node

Returns None

Raises DevFailed exception if error occurs while executing command on leaf nodes.

standby_mccs_mln (*mccs_master_fqdn*)

Create TangoClient for MccsMasterLeafNode node and call standby method.

Returns None

standby_subarray (*subarray_fqdn_list*)

Create TangoClient for Subarray node and call standby method.

Returns None

```
class tmcprototype.centralnode_low.src.centralnode_low.central_node_low.StartupTelescope(*args, **kwargs)
```

A class for Low CentralNode's StartupCommand() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do ()

Setting the startup state to TRUE enables the telescope to accept subarray commands as per the subarray model. Set the CentralNode into ON state.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (resultCode, str)

startup_leaf_node (*tango_client*)

Invoke On command on leaf nodes.

Parameters **tango_client** – Proxy of corresponding node.

Returns None

Raises DevFailed exception if error occurs while executing On command on leaf node.

startup_mccs_mln (*mccs_master_fqdn*)

Create TangoClient for MccsMasterLeafNode node and call startup method.

Returns None

startup_subarray (*subarray_fqdn_list*)

Create TangoClient for Subarray node and call startup method.

Returns None

AssignResources class for CentralNodeLow.

class tmcprototype.centralnodelow.src.centralnodelow.assign_resources_command.**AssignResources**

A class for CentralNode's AssignResources() command.

assign_resources_leaf_node (*tango_client, input_arg*)

Invokes assign Resources command on leaf node with input argument.

Parameters

- **tango_client** – client of corresponding leaf node
- **input_arg** – Json string input to invoke command.

Raises DevFailed if error occurs while invoking command on any of the devices like SubarrayNode, MCCSMasterLeafNode

assign_resources_mccs (*json_argument, device_data*)

Create proxy of mccs master leaf node and call method assign_resources_leaf_node.

Parameters **json_argument** – The string in JSON format. **device_data** : Object of class device_data.

return: None

assign_resources_subarray (*json_argument, device_data*)

Delete subarray id from json argument and create proxy of subarray corresponding to subarray id and call assign_resources_leaf_node method.

Parameters **json_argument** – The string in JSON format. **device_data** : Object of class device_data

Returns None

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (*argin*)

Assigns resources to given subarray. It accepts the subarray id, station ids, station beam id and channels in JSON string format.

Parameters **argin** – The string in JSON format. The JSON contains following values:

subarray_id: DevShort. Mandatory. Sub-Array to allocate resources to

station_ids: DevArray. Mandatory list of stations contributing beams to the data set

channels: DevArray. Mandatory list of frequency channels used

station_beam_ids: DevArray. Mandatory logical ID of beam

Example:

```
{
  "subarray_id": 1, "station_ids": [1,2], "channels": [1,2,3,4,5,6,7,8], "station_beam_ids": [1]
}
```

Note: Enter input without spaces as:{“subarray_id”:1,”station_ids”:[1,2],”channels”:[1,2,3,4,5,6,7,8],”station_beam_ids”:[1]

Returns None

Raises KeyError if input argument json string contains invalid key ValueError if input argument json string contains invalid value

StartUpTelescope class for CentralNodelow.

class tmcprototype.centralnodelow.src.centralnodelow.health_state_aggregator. **HealthStateAggregator**
Aggrerator class for health state event supscription and health state callback.

health_state_cb (evt)

Receives the subscribed Subarray health state and MCCS Master Leaf Node health state, aggregates them to calculate the telescope health state.

Parameters **evt** – A event on Subarray healthState and MCCSMasterLeafNode healthstate.

Type Event object It has the following members:

- date (event timestamp)
- reception_date (event reception timestamp)
- type (event type)
- dev_name (device name)
- name (attribute name)
- value (event value)

Returns None

Raises KeyError if error occurs while setting telescope healthState

mccs_health_subscribe_event ()

Method to subscribe to health state change event on MccsMasterLeafNode.

Raises Devfailed exception if errore occures while subscribing event.

subarray_health_subscribe_event ()

Method to subscribe to health state change event on SubarrayNode.

Raises Devfailed exception if errore occures while subscribing event.

subscribe_event ()

Method for event subscription. Calls separate subscribe event methods for CSP Master, SDP Master and Subarray health state attribute subscription.

unsubscribe_event ()

Method to unsubscribe to health state change event on MccsMasterLeafNode and SubarrayNode

ReleaseResources class for CentralNodeLow.

class tmcprototype.centralnodelow.src.centralnodelow.release_resources_command. **ReleaseResources**

A class for CentralNodeLow’s ReleaseResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (*argin*)

Release all the resources assigned to the given Subarray. It accepts the subarray id, release_all flag in JSON string format. When the release_all flag is True, ReleaseAllResources command is invoked on the respective SubarrayNode.

Parameters *argin* – The string in JSON format. The JSON contains following values:

subarray_id: DevShort. Mandatory.

release_all: Boolean(True or False). Mandatory. True when all the resources to be released from Subarray.

Example:

```
{ "subarray_id": 1, "release_all": true,  
}
```

Note: From Jive, enter input as: {“subarray_id”:1,”release_all”:true} without any space.

Raises ValueError if input argument json string contains invalid value KeyError if input argument json string contains invalid key DevFailed if the command execution or command invocation on SubarrayNode is not successful

StandByTelescope class for CentralNodelow.

```
class tmcprototype.centralnodelow.src.centralnodelow.standby_telescope_command.StandByTele
```

A class for Low CentralNode's StandByTelescope() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Sets the CentralNodeLow into OFF state. Invokes the respective command on lower level nodes and devices.

param *argin*: None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

standby_leaf_node (*tango_client*)

Invoke command Off leaf nodes.

Parameters *tango_client* – proxy of corresponding leaf node

Returns None

Raises Devfailed exception if error occurs while executing command on leaf nodes.

standby_mccs_mln (*mccs_master_fqdn*)

Create TangoClient for MccsMasterLeafNode node and call standby method.

Returns None

standby_subarray (*subarray_fqdn_list*)

Create TangoClient for Subarray node and call standby method.

Returns None

StartUpTelescope class for CentralNodelow.

class tmcprototype.centralnodelow.src.centralnodelow.startup_telescope_command.**StartUpTeles**

A class for Low CentralNode's StartupCommand() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Setting the startup state to TRUE enables the telescope to accept subarray commands as per the subarray model. Set the CentralNode into ON state.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

startup_leaf_node (*tango_client*)

Invoke On command on leaf nodes.

Parameters **tango_client** – Proxy of corresponding node.

Returns None

Raises Devfailed exception if error occurs while executing On command on leaf node.

startup_mccs_mln (*mccs_master_fqdn*)

Create TangoClient for MccsMasterLeafNode node and call startup method.

Returns None

startup_subarray (*subarray_fqdn_list*)

Create TangoClient for Subarray node and call startup method.

Returns None

CHAPTER 3

Subarray Node

Subarray Node Provides the monitoring and control interface required by users as well as other TM Components (such as OET, Central Node) for a Subarray.

```
class tmcprototype.subarraynode.src.subarraynode.subarray_node.SubarrayNode(*args,  
**kwargs)
```

Provides the monitoring and control interface required by users as well as other TM Components (such as OET, Central Node) for a Subarray.

CspSubarrayFQDN

Used by autodoc_mock_imports.

CspSubarrayLNFQDN

Used by autodoc_mock_imports.

DishLeafNodePrefix

Used by autodoc_mock_imports.

class InitCommand(*args, **kwargs)

A class for the TMC SubarrayNode's init_device() method.

do()

Initializes the attributes and properties of the Subarray Node.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the error while subscribing the tango attribute

SdpSubarrayFQDN

Used by autodoc_mock_imports.

SdpSubarrayLNFQDN

Used by autodoc_mock_imports.

Track(argin)

Invokes Track command on the Dishes assigned to the Subarray.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

calculate_observation_state()

Calculates aggregated observation state of Subarray.

command_class_object()

Sets up the command objects :return: None

delete_device()

Internal construct of TANGO.

get_deviceproxy(device_fqdn)

Returns device proxy for given FQDN.

health_state_cb(event)

Retrieves the subscribed health states, aggregates them to calculate the overall subarray health state.

Parameters **event** – A TANGO_CHANGE event on Subarray healthState.

Returns None

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_Track_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

observation_state_cb(evt)

Retrieves the subscribed CSP_Subarray AND SDP_Subarray obsState.

Parameters **evt** – A TANGO_CHANGE event on CSP and SDP Subarray obsState.

Returns None

pointing_state_cb(evt)

Retrieves the subscribed DishMaster health state, aggregate them to evaluate health state of the Subarray.

Parameters **evt** – A TANGO_CHANGE event on DishMaster healthState.

Returns None

read_activityMessage()

Internal construct of TANGO. Returns activityMessage. Example: “Subarray node is initialized successfully” //result occurred after initialization of device.

read_receptorIDList()

Internal construct of TANGO. Returns the receptor IDs allocated to the Subarray.

read_sbID()

Internal construct of TANGO. Returns the scheduling block ID.

read_scanID()

Internal construct of TANGO. Returns the Scan ID.

EXAMPLE: 123 Where 123 is a Scan ID from configuration json string.

receive_addresses_cb (event)

Retrieves the receiveAddresses attribute of SDP Subarray.

Parameters **event** – A TANGO_CHANGE event on SDP Subarray receiveAddresses attribute.

Returns None

receptorIDList

Used by autodoc_mock_imports.

remove_receptors_from_group ()

Deletes tango group of the resources allocated in the subarray.

Note: Currently there are only receptors allocated so the group contains only receptor ids.

Parameters **argin** – DevVoid

Returns DevVoid

sbID

Used by autodoc_mock_imports.

scanID

Used by autodoc_mock_imports.

validate_obs_state ()**write_activityMessage (value)**

Internal construct of TANGO. Sets the activityMessage.

```
tmcprototype.subarraynode.src.subarraynode.subarray_node.main(args=None,
                                                               **kwargs)
```

Runs the SubarrayNode. :param args: Arguments internal to TANGO :param kwargs: Arguments internal to TANGO :return: SubarrayNode TANGO object.

OnCommand class for SubarrayNode

```
class tmcprototype.subarraynode.src.subarraynode.on_command.OnCommand(*args,
                                                               **kwargs)
```

A class for the SubarrayNode's On() command.

do ()

This command invokes On Command on CSPSubarray and SDPSubarray through respective leaf nodes. This comamnd changes Subaray device state from OFF to ON.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

Raises DevFailed if the command execution is not successful

OffCommand class for SubarrayNode

```
class tmcprototype.subarraynode.src.subarraynode.off_command.OffCommand(*args,
                                                               **kwargs)
```

A class for the SubarrayNodes's Off() command.

do ()

This command invokes Off Command on CSPSubarray and SDPSubarray through respective leaf nodes. This comamnd changes Subaray device state from ON to OFF.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if the command execution is not successful

AssignResourcesCommand class for SubarrayNode.

class tmcprototype.subarraynode.src.subarraynode.assign_resources_command.**AssignResourcesC**

A class for SubarrayNode's AssignResources() command.

add_receptors_in_group (*argin*)

Creates a tango group of the successfully allocated resources in the subarray. Device proxy for each of the resources is created. The healthState and pointingState attributes from all the devices in the group are subscribed so that the changes in the respective device are received at Subarray Node.

Note: Currently there are only receptors allocated so the group contains only receptor ids.

Parameters *argin* – DevVarStringArray. List of receptor IDs to be allocated to subarray.

Example: ['0001', '0002']

Returns DevVarStringArray. List of Resources added to the Subarray. Example: ['0001', '0002']

assign_csp_resources (*argin*)

This function accepts the receptor IDs list as input and invokes the assign resources command on the CSP Subarray Leaf Node.

Parameters *argin* – List of strings Contains the list of strings that has the resources ids.

Currently this list contains only receptor ids.

Example: ['0001', '0002']

Returns List of strings. Returns the list of CSP resources successfully assigned to the Subarray. Currently, the CSPSubarrayLeafNode.AssignResources function returns void. The function only loops back the input argument in case of successful resource allocation, or returns exception object in case of failure.

assign_sdp_resources (*argin*)

This function accepts the receptor ID list as input and assigns SDP resources to SDP Subarray through SDP Subarray Leaf Node.

Parameters *argin* – List of strings Contains the list of strings that has the resources ids.

Currently processing block ids are passed to this function.

Returns

List of strings.

Example: ['PB1', 'PB2']

Returns the list of successfully assigned resources. Currently the SDPSubarrayLeafNode.AssignResources function returns void. Thus, this function just loops back the input argument in case of success or returns exception object in case of failure.

do (*argin*)

Assigns resources to the subarray. It accepts receptor id list as well as SDP resources string as a DevString. Upon successful execution, the 'receptorIDList' attribute of the subarray is updated with the list of receptors and SDP resources string is pass to SDPSubarrayLeafNode, and returns list of assigned resources as well as passed SDP string as a DevString.

Note: Resource allocation for CSP and SDP resources is also implemented but currently CSP accepts only receptorIDList and SDP accepts resources allocated to it.

Parameters *argin* – DevString.

Example:

```
{
  "dish": {
    "receptorIDList": ["0002", "0001"],
    "sdp": {
      "id": "sbi-mvp01-20200325-00001",
      "max_length": 100.0,
      "scan_types": [
        {
          "id": "science_A",
          "coordinate_system": "ICRS",
          "ra": "02:42:40.771",
          "dec": "-00:00:47.84",
          "channels": [
            {
              "count": 744,
              "start": 0,
              "stride": 2,
              "freq_min": 0.35e9,
              "freq_max": 0.368e9,
              "link_map": [[0, 0], [200, 1], [744, 2], [944, 3]]
            }
          ],
          "count": 744,
          "start": 2000,
          "stride": 1,
          "freq_min": 0.36e9,
          "freq_max": 0.368e9,
          "link_map": [[2000, 4], [2200, 5]]
        }
      ],
      "id": "calibration_B",
      "coordinate_system": "ICRS",
      "ra": "12:29:06.699",
      "dec": "02:03:08.598",
      "channels": [
        {
          "count": 744,
          "start": 0,
          "stride": 2,
          "freq_min": 0.35e9,
          "freq_max": 0.368e9,
          "link_map": [[0, 0], [200, 1], [744, 2], [944, 3]],
          "start": 2000,
          "stride": 1,
          "freq_min": 0.36e9,
          "freq_max": 0.368e9,
          "link_map": [[2000, 4], [2200, 5]]
        }
      ],
      "processing_blocks": [
        {
          "id": "pb-mvp01-20200325-00001",
          "workflow": [
            {
              "type": "realtime",
              "id": "vis_receive",
              "version": "0.1.0"
            },
            {
              "parameters": {}
            }
          ],
          "id": "pb-mvp01-20200325-00002",
          "workflow": [
            {
              "type": "realtime",
              "id": "test_realtime",
              "version": "0.1.0"
            },
            {
              "parameters": {}
            }
          ],
          "id": "pb-mvp01-20200325-00003",
          "workflow": [
            {
              "type": "batch",
              "id": "ical",
              "version": "0.1.0"
            },
            {
              "parameters": {}
            }
          ],
          "dependencies": [
            {
              "pb_id": "pb-mvp01-20200325-00001",
              "type": "visibilities"
            }
          ],
          "id": "pb-mvp01-20200325-00004",
          "workflow": [
            {
              "type": "batch",
              "id": "dpreb",
              "version": "0.1.0"
            },
            {
              "parameters": {}
            }
          ],
          "dependencies": [
            {
              "pb_id": "pb-mvp01-20200325-00003",
              "type": "calibration"
            }
          ]
        }
      ]
    }
  }
}
```

Returns

A tuple containing a return code and string of Resources added to the Subarray. Example of string of Resources :

```
["0001", "0002"]
```

as argout if allocation successful.

Return type (resultCode, str)

Raises ValueError if input argument json string contains invalid value DevFailed if the command execution is not successful

ReleaseAllResourcesCommand for SubarrayNode

```
class tmcprototype.subarraynode.src.subarraynode.release_all_resources_command.ReleaseAllRe
```

A class for SKASubarray's ReleaseAllResources() command.

do()

It checks whether all resources are already released. If yes then it throws error while executing command. If not it Releases all the resources from the subarray i.e. Releases resources from TMC Subarray Node, CSP Subarray and SDP Subarray. If the command execution fails, array of receptors(device names) which are failed to be released from the subarray, is returned to Central Node. Upon successful execution, all the resources of a given subarray get released and empty array is returned. Selective release is not yet supported.

Returns A tuple containing a return code and "[]" as a string on successful release all resources.

Example: "[]" as string on successful release all resources.

Return type (resultCode, str)

Raises DevFailed if the command execution is not successful

release_csp_resources()

This function invokes releaseAllResources command on CSP Subarray via CSP Subarray Leaf Node.

Parameters **argin** – DevVoid

Returns DevVoid

release_sdp_resources()

This function invokes releaseAllResources command on SDP Subarray via SDP Subarray Leaf Node.

Parameters **argin** – DevVoid

Returns DevVoid

ConfigureCommand class for SubarrayNode.

class tmcprototype.subarraynode.src.subarraynode.configure_command.**ConfigureCommand**(*args,
**kwargs)

A class for SubarrayNode's Configure() command.

do (argin)

Configures the resources assigned to the Subarray. The configuration data for SDP, CSP and Dish is extracted out of the input configuration string and relayed to the respective underlying devices (SDP Subarray Leaf Node, CSP Subarray Leaf Node and Dish Leaf Node).

Parameters **argin** – DevString.

JSON string that includes pointing parameters of Dish - Azimuth and Elevation Angle, CSP Configuration and SDP Configuration parameters. JSON string example is: {“pointing”:{“target”:{“system”:“ICRS”,“name”:“Polaris Australis”,“RA”:“21:08:47.92”,“dec”:-88:57:22.9}},“dish”:{“receiverBand”：“1”,“csp”:{“id”：“sbi-mvp01-20200325-00001-science_A”,“frequencyBand”：“1”,“fsp”:[{‘fspID’:1,‘functionMode’:“CORR”,‘frequencySliceID’:1,‘integrationTime’:14},“sdp”:{“scan_type”:“science_A”},“tmc”:{“scanDuration”:10.0}}}} CSP block in json string is as per earlier implementation and not aligned to SP-872 Note: While invoking this command from JIVE, provide above JSON string without any space.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ReturnCode, str)

Raises JSONDecodeError if input argument json string contains invalid value

class tmcprototype.subarraynode.src.subarraynode.configure_command.**ElementDeviceData**

static build_up_csp_cmd_data (scan_config, attr_name_map, receive_addresses_map)

Here the input data for CSP is build which is used in configuration of CSP. Below is the csp_config_schema variable value generated by using ska_telmodel library. {‘id’: ‘sbi-mvp01-20200325-00001-science_A’, ‘frequencyBand’: ‘1’, ‘fsp’: [{‘fspID’: 1, ‘functionMode’ : ‘CORR’, ‘frequencySliceID’: 1, ‘integrationTime’: 1400, ‘corrBandwidth’: 0, ‘channelAveragingMap’: [[0, 2], [744, 0]], ‘fspChannelOffset’: 0, ‘outputLinkMap’: [[0, 0], [200, 1]], ‘outputHost’: [[0, ‘192.168.0.1’], [400, ‘192.168.0.2’]], ‘outputMac’: [[0, ‘06-00-00-00-00-00’]], ‘outputPort’: [[0, 9000, 1], [400, 9000, 1]]}, {‘fspID’: 2, ‘functionMode’: ‘CORR’, ‘frequencySliceID’: 2, ‘integrationTime’: 1400, ‘corrBandwidth’: 0, ‘channelAveragingMap’: [[0, 2], [744, 0]], ‘fspChannelOffset’: 744, ‘outputLinkMap’: [[0, 4], [200, 5]], ‘outputHost’: [[0, ‘192.168.0.3’], [400, ‘192.168.0.4’]], ‘outputMac’: [[0, ‘06-00-00-00-00-01’]], ‘outputPort’: [[0, 9000, 1], [400, 9000, 1]]}]}

Returns csp configuration schema

static build_up_dsh_cmd_data (scan_config, only_dishconfig_flag)

static build_up_sdp_cmd_data (scan_config)

ScanCommand class for SubarrayNode

class tmcprototype.subarraynode.src.subarraynode.scan_command.**ScanCommand**(*args,
**kwargs)

A class for SubarrayNode's Scan() command.

```
call_end_scan_command()
```

do (argin)

This command accepts id as input. And it Schedule scan on subarray from where scan command is invoked on respective CSP and SDP subarray node for the provided interval of time. It checks whether the scan is already in progress. If yes it throws error showing duplication of command.

Parameters **argin** – DevString. JSON string containing id.

JSON string example as follows:

```
{"id": 1}
```

Note: Above JSON string can be used as an input argument while invoking this command from JIVE.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the command execution is not successful

EndScanCommand class for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.end_scan_command.EndScanCommand(*args,  
**kwargs)
```

A class for SubarrayNode's EndScan() command.

do ()

Ends the scan. It is invoked on subarray after completion of the scan duration. It can also be invoked by an external client while a scan is in progress, Which stops the scan immediately irrespective of the provided scan duration.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the command execution is not successful.

EndCommand class for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.end_command.EndCommand(*args,  
**kwargs)
```

A class for SubarrayNode's End() command.

do ()

This command on Subarray Node invokes EndSB command on CSP Subarray Leaf Node and SDP Subarray Leaf Node, and stops tracking of all the assigned dishes.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if the command execution is not successful.

```
stop_dish_tracking()
```

AbortCommand for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.abort_command.AbortCommand(*args,  
**kwargs)
```

A class for SubarrayNode's Abort() command.

do ()

This command on Subarray Node invokes Abort command on CSP Subarray Leaf Node and SDP Subarray Leaf Node, and stops tracking of all the assigned dishes.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

Raises DevFailed if error occurs in invoking command on any of the devices like CSPSubarrayLeafNode, SDPSubarrayLeafNode or DishLeafNode

RestartCommand for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.restart_command.RestartCommand(*args,  
**kwargs)
```

A class for SubarrayNode's Restart() command.

do ()

This command invokes Restart command on CSPSubarrayLeafNode, SDpSubarrayLeafNode and DishLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

Raises DevFailed if error occurs while invoking command on CSPSubarrayLeafNode, SDpSubarrayLeafNode or DishLeafNode.

ObsResetCommand for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.obsreset_command.ObsResetCommand(*args,  
**kwargs)
```

A class for SubarrayNode's ObsReset() command.

do ()

This command invokes ObsReset command on CspSubarrayLeafNode, SdpSubarrayLeafNode and DishLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

Raises DevFailed if error occurs while invoking command on CspSubarrayLeafNode, SdpSubarrayLeafNode or DishLeafNode.

TrackCommand class for SubarrayNode.

```
class tmcprototype.subarraynode.src.subarraynode.track_command.TrackCommand(*args,  
**kwargs)
```

A class for SubarrayNode's Track command.

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do (argin)

Invokes Track command on the Dishes assigned to the Subarray.

Parameters `argin` – DevString

Example: radec|21:08:47.92|-88:57:22.9 as argin Argin to be provided is the Ra and Dec values where first value is tag that is radec, second value is Ra in Hr:Min:Sec, and third value is Dec in Deg:Min:Sec.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

CHAPTER 4

Subarray Node

Subarray Node Low Provides the monitoring and control interface required by users as well as other TM Components (such as OET, Central Node) for a Subarray.

```
class tmcprototype.subarraynodelow.src.subarraynodelow.subarray_node_low.SubarrayNode(*args,  
**kwargs)
```

Provides the monitoring and control interface required by users as well as other TM Components (such as OET, Central Node) for a Subarray.

```
class InitCommand(*args, **kwargs)
```

A class for the TMC SubarrayNode's init_device() method.

```
do()
```

Initializes the attributes and properties of the Subarray Node.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the error while subscribing the tango attribute

MccsSubarrayFQDN

Used by autodoc_mock_imports.

MccsSubarrayLNFQDN

Used by autodoc_mock_imports.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

calculate_observation_state()

Calculates aggregated observation state of Subarray.

command_class_object()

Sets up the command objects :return: None

delete_device()

Internal construct of TANGO.

get_deviceproxy (device_fqdn)

Returns device proxy for given FQDN.

health_state_cb (event)

Receives the subscribed health states, aggregates them to calculate the overall subarray health state.

Parameters **evt** – A event on MCCS Subarray healthState.

Type Event object It has the following members:

- date (event timestamp)
- reception_date (event reception timestamp)
- type (event type)
- dev_name (device name)
- name (attribute name)
- value (event value)

Returns None

init_command_objects()

Initialises the command handlers for commands supported by this device.

observation_state_cb (evt)

Receives the subscribed MCCS Subarray obsState.

Parameters **evt** – A event on MCCS Subarray ObsState.

Type Event object It has the following members:

- date (event timestamp)
- reception_date (event reception timestamp)
- type (event type)
- dev_name (device name)
- name (attribute name)
- value (event value)

Returns None

Raises KeyError if error occurs while setting SubarrayNode's ObsState.

read_activityMessage()

Internal construct of TANGO. Returns activityMessage. Example: “Subarray node is initialized successfully” //result occurred after initialization of device.

read_scanID()

Internal construct of TANGO. Returns the Scan ID.

EXAMPLE: 123 Where 123 is a Scan ID from configuration json string.

scanID

Used by autodoc_mock_imports.

write_activityMessage (value)

Internal construct of TANGO. Sets the activityMessage.

```
tmcprototype.subarraynode_low.src.subarraynode_low.subarray_node_low.main(args=None,
                                                               **kwargs)
```

Runs the SubarrayNode. :param args: Arguments internal to TANGO :param kwargs: Arguments internal to TANGO :return: SubarrayNode TANGO object.

AssignResourcesCommand class for SubarrayNodeLow.

```
class tmcprototype.subarraynode_low.src.subarraynode_low.assign_resources_command.AssignResou
```

A class for SubarrayNodelow's AssignResources() command.

do (argin)

Assigns the resources to the subarray. It accepts station ids, channels, station beam ids

Parameters argin – DevString in JSON form containing following fields: station_ids: list of integers

channels: list of integers

station_beam_ids: list of integers

Example:

```
{“station_ids”: [1, 2], “channels”: [1, 2, 3, 4, 5, 6, 7, 8], “station_beam_ids”: [1]}
```

Returns A tuple containing ResultCode and string.

ConfigureCommand class for SubarrayNodeLow.

```
class tmcprototype.subarraynode_low.src.subarraynode_low.configure_command.ConfigureCommand(
```

A class for SubarrayNodeLow's Configure() command.

do (argin)

Configures the resources assigned to the Mccs Subarray.

Parameters argin – DevString.

JSON string example is:

```
{“mccs”:{“stations”:[{“station_id”:1},{“station_id”:2}],“station_beam_pointings”: [{“sta-  
tion_beam_id”:1,“target”:{“system”:“HORIZON”,“name”:“DriftScan”,“Az”:180.0,“El”:45.0},  
“update_rate”:0.0,“channels”:[1,2,3,4,5,6,7,8]}]},“tmc”:{“scanDuration”:10.0}}
```

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ReturnCode, str)

Raises JSONDecodeError if input argument json string contains invalid value DevFailed if the command execution is not successful.

EndCommand class for SubarrayNodeLow.

```
class tmcprototype.subarraynode_low.src.subarraynode_low.end_command.EndCommand(*args,  
                                                               **kwargs)
```

A class for SubarrayNodeLow's End() command.

do ()

This command on Subarray Node Low invokes End command on MCCS Subarray Leaf Node.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if the command execution is not successful.

EndScanCommand class for SubarrayNodeLow.

```
class tmcprototype.subarraynodelow.src.subarraynodelow.end_scan_command.EndScanCommand(*args, **kwargs)
```

A class for SubarrayNodeLow's EndScan() command.

do()

Ends the scan. It is invoked on subarrayLow after completion of the scan duration. It can also be invoked by an external client while a scan is in progress, Which stops the scan immediately irrespective of the provided scan duration.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the command execution is not successful.

OffCommand class for SubarrayNodeLow

```
class tmcprototype.subarraynodelow.src.subarraynodelow.off_command.OffCommand(*args, **kwargs)
```

A class for the SubarrayNodes's Off() command.

do()

This command invokes Off Command on MCCSSubarray through mcs subarray leaf node. This command changes Subarray device state from ON to OFF.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if the command execution is not successful

OnCommand class for SubarrayNodeLow

```
class tmcprototype.subarraynodelow.src.subarraynodelow.on_command.OnCommand(*args, **kwargs)
```

A class for the SubarrayNodeLow's On() command.

do()

This command invokes On Command on MCCSSubarray through MCCS Subarray Leaf node. This command changes Subarray device state from OFF to ON.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if the command execution is not successful

ReleaseAllResourcesCommand for SubarrayNodeLow

```
class tmcprototype.subarraynodelow.src.subarraynodelow.release_all_resources_command.ReleaseAllResourcesCommand(*args, **kwargs)
```

A class for SKASubarrayLow's ReleaseAllResources() command.

do()

It invokes ReleaseAllResources command on Subarraylow.

Returns A tuple containing a return code and RELEASEALLRESOURCES command invoked successfully as a string on successful release all resources.

Example: RELEASEALLRESOURCES command invoked successfully as string on successful release all resources.

Return type (ResultCode, str)

ScanCommand class for SubarrayNodeLow

```
class tmcprototype.subarraynodelow.src.subarraynodelow.scan_command.ScanCommand(*args,  
**kwargs)
```

A class for SubarrayNodeLow's Scan() command.

```
call_end_scan_command()
```

```
do (argin)
```

This command accepts id as input. And it Schedule scan on subarray from where scan command is invoked on M CCS subarray Leaf Node for the provided interval of time. It checks whether the scan is already in progress. If yes it throws error showing duplication of command.

Parameters **argin** – DevString. JSON string containing id.

JSON string example as follows:

```
{"id": 1}
```

Note: Above JSON string can be used as an input argument while invoking this command from JIVE.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the command execution is not successful

CHAPTER 5

Dish Leaf Node

A Leaf control node for DishMaster.

```
class tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode(*args,  
**kwargs)
```

A Leaf control node for DishMaster.

Abort()

Invokes Abort command on the DishMaster.

```
class AbortCommand(*args, **kwargs)
```

A class for DishLeafNode's Abort command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

do()

Invokes TrackStop command on the DishMaster.

Raises DevFailed – If error occurs while invoking TrackStop command on Dish-Master.

Configure(argin)

Configures the Dish by setting pointing coordinates for a given observation.

```
class ConfigureCommand(*args, **kwargs)
```

A class for DishLeafNode's Configure() command.

check_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state

Return type boolean

do (argin)

Configures the Dish by setting pointing coordinates for a given scan. This function accepts the input json and calculate pointing parameters of Dish- Azimuth and Elevation Angle. Calculated parameters are again converted to json and fed to the dish master.

Parameters **argin** – A String in a JSON format that includes pointing parameters of Dish- Azimuth and Elevation Angle.

Example: {“pointing”:{“target”:{“system”：“ICRS”,“name”：“Polaris Australis”,“RA”：“21:08:47.92”,“dec”：“-88:57:22.9”}},
“dish”:{“receiverBand”：“1”}}

Raises **DevFailed** – If error occurs while invoking ConfigureBand<> command on DishMaster or if the json string contains invalid data.

DishMasterFQDN

Used by autodoc_mock_imports.

EndScan (argin)

Invokes StopCapture command on DishMaster.

class EndScanCommand (*args, **kwargs)

A class for DishLeafNode’s EndScan() command.

check_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)

Invokes StopCapture command on DishMaster.

Parameters **argin** – timestamp

Raises **DevFailed** – If error occurs while invoking StopCapture command on Dish-Master.

class InitCommand (*args, **kwargs)

A class for the TMC DishLeafNode’s init_device() method.

do ()

Initializes the attributes and properties of the DishLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

Raises **DevFailed** – If error occurs in creating a DeviceProxy instance for DishMas-ter

ObsReset ()

Invokes ObsReset command on the DishLeafNode.

class ObsResetCommand (*args, **kwargs)

A class for DishLeafNode’s ObsReset command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

do ()
Invokes StopCapture command on the DishMaster.

Raises DevFailed – If error occurs while invoking StopCapture command on DishMaster.

Restart ()
Invokes Restart command on the DishMaster.

class RestartCommand(*args, **kwargs)
A class for DishLeafNode's Restart command.

check_allowed()
Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

do ()
Invokes StopCapture command on the DishMaster.

Raises DevFailed – If error occurs while invoking StopCapture command on DishMaster.

Scan (argin)
Invokes Scan command on DishMaster.

class ScanCommand(*args, **kwargs)
A class for DishLeafNode's Scan() command.

check_allowed()
Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)
Invokes Scan command on DishMaster.

Parameters **argin** – timestamp

Raises DevFailed – If error occurs while invoking Scan command on DishMaster.

SetOperateMode ()
Invokes SetOperateMode command on DishMaster.

class SetOperateModeCommand(*args, **kwargs)
A class for DishLeafNode's SetOperateMode() command.

do ()
Invokes SetOperateMode command on DishMaster.

Raises DevFailed – If error occurs while invoking SetOperateMode command on DishMaster.

SetStandbyFPMode ()
Invokes SetStandbyFPMode command on DishMaster (Standby-Full power) mode.

class SetStandbyFPModeCommand(*args, **kwargs)
A class for DishLeafNode's SetStandbyFPMode() command.

do ()
Invokes SetStandbyFPMode command on DishMaster (Standby-Full power) mode.

Raises DevFailed – If error occurs while invoking SetStandbyFPMode command on DishMaster.

SetStandbyLPMode ()

Invokes SetStandbyLPMode (i.e. Low Power State) command on DishMaster.

class SetStandbyLPModeCommand (*args, **kwargs)

A class for DishLeafNode's SetStandbyLPMode() command.

do ()

Invokes SetStandbyLPMode (i.e. Low Power State) command on DishMaster.

Raises DevFailed – If error occurs while invoking SetStandbyLPMode command on DishMaster.

SetStowMode ()

Invokes SetStowMode command on DishMaster.

class SetStowModeCommand (*args, **kwargs)

A class for DishLeafNode's SetStowMode() command.

do ()

Invokes SetStowMode command on DishMaster.

Raises DevFailed – If error occurs while invoking SetStowMode command on DishMaster.

Slew (argin)

Invokes Slew command on DishMaster to slew the dish towards the set pointing coordinates.

class SlewCommand (*args, **kwargs)

A class for DishLeafNode's SlewCommand() command.

check_allowed ()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)

Invokes Slew command on DishMaster to slew the dish towards the set pointing coordinates.

Parameters **argin** – list [0] = Azimuth, in degrees [1] = Elevation, in degrees

Raises DevFailed – If error occurs while invoking Slew command on DishMaster.

StartCapture (argin)

Triggers the DishMaster to Start capture on the set configured band.

class StartCaptureCommand (*args, **kwargs)

A class for DishLeafNode's StartCapture() command.

check_allowed ()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)

Invokes StartCapture command on DishMaster on the set configured band.

Parameters **argin** – timestamp

Raises DevFailed – If error occurs while invoking StartCapture command on Dish-Master.

StopCapture (argin)

Invokes StopCapture command on DishMaster on the set configured band.

class StopCaptureCommand (*args, **kwargs)

A class for DishLeafNode's StopCapture() command.

check_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)

Invokes StopCapture command on DishMaster on the set configured band.

Parameters **argin** – timestamp

Raises DevFailed – If error occurs while invoking StopCapture command on Dish-Master.

StopTrack ()

Invokes StopTrack command on the DishMaster.

class StopTrackCommand (*args, **kwargs)

A class for DishLeafNode's StopTrack() command.

check_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do ()

Invokes TrackStop command on the DishMaster.

Raises DevFailed – If error occurs while invoking TrackStop command on Dish-Master.

Track (argin)

Invokes Track command on the DishMaster.

class TrackCommand (*args, **kwargs)

A class for DishLeafNode's Track() command.

check_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

do (argin)

Invokes Track command on the DishMaster.

Parameters **argin** – DevString The elevation limit thread allows Dish to track a source till the observation capacity i.e. elevation limit of dish.

The tracking time thread allows dish to track a source for the prespecified Track Duration (provided elevation limit is not reached).

For Track command, argin to be provided is the Ra and Dec values in the following JSON format:

```
{"pointing":{ "target":{ "system":"ICRS","name":"Polaris Australis","RA": "21:08:47.92","dec": "-88:57:22.9" }},  
 "dish":{ "receiverBand": "1" }}
```

Raises DevFailed – If error occurs while invoking Track command on DishMaster.

activityMessage

Used by autodoc_mock_imports.

attribute_event_handler(event_data)

Retrieves the subscribed attribute of DishMaster.

Parameters **evt** – A TANGO_CHANGE event on attribute.

convert_radec_to_azel(target, timestamp)

Converts RaDec coordinate in to AzEl coordinate using KATPoint library.

Parameters

- **target** – str Argin to be provided is the Ra and Dec values in the following format: radec,21:08:47.92,89:15:51.4
- **timestamp** – str 2020-12-11 10:06:34.970731

Returns list Azimuth and elevation angle, in degrees

Raises ValueError – If error occurs when creating katpoint Target or Timestamp.

dishHealthState

Used by autodoc_mock_imports.

dishPointingState

Used by autodoc_mock_imports.

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_Abort_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_Configure_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_EndScan_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_ObsReset_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_Restart_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_Scan_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_Slew_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_StartCapture_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_StopCapture_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_StopTrack_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_Track_allowed()

Checks whether this command is allowed to be run in the current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

read_activityMessage()

Returns the activityMessage

set_dish_name_number()**set_observer_lat_long_alt()****track_thread()**

This thread writes coordinates to desiredPointing on DishMaster at the rate of 20 Hz.

write_activityMessage (value)

Internal construct of TANGO. Sets the activityMessage

```
tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.main(args=None,  
**kwargs)
```

Runs the DishLeafNode. :param args: Arguments internal to TANGO :param kwargs: Arguments internal to TANGO :return: DishLeafNode TANGO object.

CHAPTER 6

Dish Master

CHAPTER 7

CSP Master Leaf Node

CSP Master Leaf node monitors the CSP Master and issues control actions during an observation.

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
```

Properties:

- CspMasterFQDN - Property to provide FQDN of CSP Master Device

Attributes:

- cspHealthState - Forwarded attribute to provide CSP Master Health State # - activityMessage - Attribute to provide activity message

#

CspMasterFQDN

Used by autodoc_mock_imports.

class InitCommand(*args, **kwargs)

A class for the TMC CSP Master Leaf Node's init_device() method.

do()

Initializes the attributes and properties of the CspMasterLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if error occurs while creating the device proxy for CSP Master or subscribing the evennts.

Standby(argin)

Sets Standby Mode on the CSP Element.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

cspHealthState

Used by autodoc_mock_imports.

delete_device()

Internal construct of TANGO.

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_Standby_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

read_activityMessage()

Internal construct of TANGO. Returns the activityMessage.

write_activityMessage(value)

Internal construct of TANGO. Sets the activityMessage.

tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.**main**(*args=None*,
***kwargs*)

Runs the CspMasterLeafNode.

Parameters

- **args** – Arguments internal to TANGO

- **kwargs** – Arguments internal to TANGO

Returns CspMasterLeafNode TANGO object.

class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.**On**(**args*,
***kwargs*)

A class for CspMasterLeafNode's On() command.

do()

Invokes On command on the CSP Element.

Parameters **argin** – None

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed on communication failure with CspMaster or CspMaster is in error state.

on_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the On command has been successfully invoked on CSPMaster.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout

- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.Off(*args,  
**kwargs)
```

A class for CspMasterLeafNode's Off() command.

do ()

Invokes Off command on the CSP Element.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.Standby(*args,  
**kwargs)
```

A class for CspMasterLeafNode's Standby() command.

check_allowed ()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

do (argin)

It invokes the STANDBY command on CSP Master. :param argin: DevStringArray. If the array length is 0, the command applies to the whole CSP Element. If the array length is > 1 , each array element specifies the FQDN of the CSP SubElement to put in STANDBY mode. :return: None :raises: DevFailed on communication failure with CspMaster or CspMaster is in error state.

standby_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the StandBy command has been successfully invoked on CSPMaster.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_command.Off (*args,  
                           **kwargs)
```

A class for CspMasterLeafNode's Off() command.

do ()

Invokes Off command on the CSP Element.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.on_command.On (*args,  
                           **kwargs)
```

A class for CspMasterLeafNode's On() command.

do ()

Invokes On command on the CSP Element.

Parameters **argin** – None

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (resultCode, str)

Raises DevFailed on communication failure with CspMaster or CspMaster is in error state.

on_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the On command has been successfully invoked on CSPMaster.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.cspmasterleafnode.src.cspmasterleafnode.standby_command.Standby (*args,  
                           **kwargs)
```

A class for CspMasterLeafNode's Standby() command.

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

do (argin)

It invokes the STANDBY command on CSP Master. :param argin: DevStringArray. If the array length is 0, the command applies to the whole CSP Element. If the array length is > 1 , each array element specifies the FQDN of the CSP SubElement to put in STANDBY mode. :return: None :raises: DevFailed on communication failure with CspMaster or CspMaster is in error state.

standby_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the StandBy command has been successfully invoked on CSPMaster.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the call-back method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

CHAPTER 8

SDP Subarray Leaf Node

CHAPTER 9

Subarray Node

SDP Subarray Leaf node is to monitor the SDP Subarray and issue control actions during an observation. It also acts as a SDP contact point for Subarray Node for observation execution.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode:
```

SDP Subarray Leaf node is to monitor the SDP Subarray and issue control actions during an observation.

Abort ()

Invoke Abort on SdpSubarrayLeafNode.

AssignResources (argin)

Assigns resources to given SDP subarray.

Configure (argin)

Invokes Configure on SdpSubarrayLeafNode.

End ()

This command invokes End command on SDP subarray to end the current Scheduling block.

EndScan ()

Invokes EndScan on SdpSubarrayLeafNode.

class InitCommand (*args, **kwargs)

A class for the TMC SdpSubarrayLeafNode's init_device() method.

do ()

Initializes the attributes and properties of the SdpSubarrayLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

ObsReset ()

Invoke ObsReset command on SdpSubarrayLeafNode.

ReleaseAllResources ()

Invokes ReleaseAllResources command on SdpSubarrayLeafNode.

Restart()

Invoke Restart command on SdpSubarrayLeafNode.

Scan(*argin*)

Invoke Scan command to SDP subarray.

SdpSubarrayFQDN

Used by autodoc_mock_imports.

activeProcessingBlocks

Used by autodoc_mock_imports.

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

delete_device()

Internal construct of TANGO.

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_Abort_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_AssignResources_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_Configure_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_EndScan_allowed()

Checks whether this command is allowed to be run in current device state. :return: True if this command is allowed to be run in current device state. :rtype: boolean

is_End_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

is_ObsReset_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_ReleaseAllResources_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_Restart_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_Scan_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

read_activeProcessingBlocks()

Internal construct of TANGO. Returns Active Processing Blocks.activeProcessingBlocks is a forwarded attribute from SDP Subarray which depicts the active Processing Blocks in the SDP Subarray

read_activityMessage()

Internal construct of TANGO. Returns Activity Messages. activityMessage is a String providing information about the current activity in SDP Subarray Leaf Node

read_receiveAddresses()

Internal construct of TANGO. Returns the Receive Addresses. receiveAddresses is a forwarded attribute from SDP Master which depicts State of the SDP.

receiveAddresses

Used by autodoc_mock_imports.

sdpSubarrayHealthState

Used by autodoc_mock_imports.

sdpSubarrayObsState

Used by autodoc_mock_imports.

validate_obs_state()**write_activityMessage (value)**

Internal construct of TANGO. Sets the Activity Message. activityMessage is a String providing information about the current activity in SDP Subarray Leaf Node.

write_receiveAddresses (value)

Internal construct of TANGO. Sets the Receive Addresses. receiveAddresses is a forwarded attribute from SDP Master which depicts State of the SDP.

```
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.main(args=No
**kwargs)
```

Runs the SdpSubarrayLeafNode

Parameters

- **args** – Arguments internal to TANGO
- **kwargs** – Arguments internal to TANGO

Returns SdpSubarrayLeafNode TANGO object

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.AssignResources
```

A class for SdpSubarrayLeafNode's AssignResources() command.

AssignResources_ended (event)

This is the callback method of AssignResources command of the SDP Subarray. It checks whether the AssignResources command on SDP subarray is successful.

Parameters argin – event: response from SDP Subarray for the invoked assign resource command.

Returns None

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

do (argin)

Assigns resources to given SDP subarray. This command is provided as a noop placeholder from SDP subarray. Eventually this will likely take a JSON string specifying the resource request.

Parameters argin – The string in JSON format. The JSON contains following values:

SBI ID and maximum length of the SBI: Mandatory JSON object consisting of

SBI ID : String

max_length: Float

Scan types: Consist of Scan type id name

scan_type: DevVarStringArray

Processing blocks: Mandatory JSON object consisting of

processing_blocks: DevVarStringArray

Example: `{"id":"sbi-mvp01-20200325-00001","max_length":100.0,"scan_types":[{"id":"science_A","coordinate_system":"ICRS","ra": "02:42:40.771","dec": "-00:00:47.84","channels":[{"count":744,"start":0,"stride":2,"freq_min":0.35e9,"freq_max":0.368e9,"link_map":[[0,0],[200,1],[744,2],[944,3]]},{“count”:744,”start”:2000,”stride”:1,”freq_min”:0.36e9,”freq_max”:0.368e9,”link_map”:[[2000,4],[2200,5]]}]}],{"id":"calibration_B","coordinate_system":"ICRS","ra": "12:29:06.699","dec": "02:03:08.598","channels":[{"count":744,"start":0,"stride":2,"freq_min":0.35e9,"freq_max":0.368e9,"link_map":[[0,0],[200,1],[744,2],[944,3]]}],{"count":744,"start":2000,"stride":1,"freq_min":0.36e9,"freq_max":0.368e9,"link_map":[[2000,4],[2200,5]]}}],"processing_blocks":[{"id":"pb-mvp01-20200325-00001","workflow":{"type":"realtime","id": "vis_receive","version": "0.1.0"}, "parameters":{}}, {"id": "pb-mvp01-20200325-00002","workflow": {"type": "realtime", "id": "test_realtime", "version": "0.1.0"}, "parameters": {}}, {"id": "pb-mvp01-20200325-00003", "workflow": {"type": "batch", "id": "ical", "version": "0.1.0"}, "parameters": {}}, {"dependencies": [{"pb_id": "pb-mvp01-20200325-00001", "type": ["visibilities"]}], {"id": "pb-mvp01-20200325-00004", "workflow": {"type": "batch", "id": "dpreb", "version": "0.1.0"}, "parameters": {}}, {"dependencies": [{"pb_id": "pb-mvp01-20200325-00003", "type": ["calibration"]}]}]}`

Note: Enter input without spaces

Returns None

Raises ValueError if input argument json string contains invalid value. DevFailed if the command execution is not successful.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.ReleaseAllResources
```

A class for SdpSubarrayLeafNode's ReleaseAllResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises Exception if command execution throws any type of exception

do()

Releases all the resources of given SDPSubarrayLeafNode. It accepts the subarray id, releaseALL flag and receptorIDList in JSON string format.

Parameters **argin** – None.

Returns None

Raises DevFailed if the command execution is not successful.

releaseallresources_cmd-ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the releaseallresources command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- **device** : (DeviceProxy) The DeviceProxy object on which the call was executed.
- **cmd_name** : (str) The command name
- **argout_raw** : (DeviceData) The command argout
- **argout** : The command argout
- **err** : (bool) A boolean flag set to true if the command failed. False otherwise
- **errors** : (sequence<DevError>) The error stack
- **ext**

Returns none

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.On (*  
**
```

A class for SDP Subarray's On() command.

do()

Invokes On command on the SDP Subarray.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

on_cmd_ended_cb (event)

Callback function executes when the command invoked asynchronously returns from the server.

Parameters event – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**Off** (

A class for SDP Subarray's Off() command.

do ()

Invokes Off command on the SDP Subarray.

Parameters argin – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

off_cmd_ended_cb (event)

Callback function executes when the command invoked asynchronously returns from the server.

Parameters event – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**Configure**

A class for SdpSubarrayLeafNode's Configure() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises Exception if command execution throws any type of exception

configure_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the configure command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- **device** : (DeviceProxy) The DeviceProxy object on which the call was executed.
- **cmd_name** : (str) The command name
- **argout_raw** : (DeviceData) The command argout
- **argout** : The command argout
- **err** : (bool) A boolean flag set to true if the command failed. False otherwise
- **errors** : (sequence<DevError>) The error stack
- **ext**

Returns none

do (*argin*)

Configures the SDP Subarray device by providing the SDP PB configuration needed to execute the receive workflow

Parameters **argin** – The string in JSON format. The JSON contains following values:

Example:

```
{ "scan_type": "science_A" }
```

Returns None

Raises ValueError if input argument json string contains invalid value. KeyError if input argument json string contains invalid key. DevFailed if the command execution is not successful

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**Abort**

A class for sdpSubarrayLeafNode's Abort() command.

abort_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the abort command has been successfully invoked on SDP Subarray.

Parameters `event` – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- `device` : (DeviceProxy) The DeviceProxy object on which the call was executed.
- `cmd_name` : (str) The command name
- `argout_raw` : (DeviceData) The command argout
- `argout` : The command argout
- `err` : (bool) A boolean flag set to true if the command failed. False otherwise
- `errors` : (sequence<DevError>) The error stack
- `ext`

Returns none

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to abort the current operation being done on the SDP subarray.

Returns None

Raises DevFailed if error occurs while invoking command on CSPSubarray.

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**Restart**

A class for sdpSubarrayLeafNode's Restart() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to restart the SDP subarray and bring it to its ON state.

Returns None

Raises DevFailed if error occurs while invoking command on SDPSubarray.

restart_cmd_ended_cb (`event`)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the restart command has been successfully invoked on SDP Subarray.

Parameters `event` – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.ObsResetCommand
```

A class for SdpSubarrayLeafNode's ObsResetCommand() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to reset the SDP subarray and bring it to its RESETTING state.

Parameters **argin** – None

Returns None

Raises DevFailed if error occurs while invoking command on SDPSubarray.

obsreset_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the ObsResetCommand has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise

- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**Scan**

A class for SdpSubarrayLeafNode's Scan() command.

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

do (argin)

Invoke Scan command to SDP subarray.

Parameters **argin** – The string in JSON format. The JSON contains following values:

Example: {“id”:1}

Note: Enter input as without spaces:{“id”:1}

Returns None

Raises DevFailed if the command execution is not successful.

scan_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the scan command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.**End**(

A class for SdpSubarrayLeafNode's End() command.

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

do()

This command invokes End command on SDP subarray to end the current Scheduling block.

Returns None

Raises DevFailed if the command execution is not successful.

end_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the end command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- **device** : (DeviceProxy) The DeviceProxy object on which the call was executed.
- **cmd_name** : (str) The command name
- **argout_raw** : (DeviceData) The command argout
- **argout** : The command argout
- **err** : (bool) A boolean flag set to true if the command failed. False otherwise
- **errors** : (sequence<DevError>) The error stack
- **ext**

Returns none

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.EndScan

A class for SdpSubarrayLeafNode's EndScan() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises Exception if command execution throws any type of exception.

do()

It invokes EndScan command on SdpSubarray. This command is allowed when SdpSubarray is in SCANNING state.

Parameters **argin** – None

Returns None

Raises DevFailed if the command execution is not successful.

endscan_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the endscan command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

On class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.On (*args,  
**kwargs)
```

A class for SDP Subarray's On() command.

do ()

Invokes On command on the SDP Subarray.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (resultCode, str)

on_cmd_ended_cb (event)

Callback function executes when the command invoked asynchronously returns from the server.

Parameters **event** – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack

- ext

Returns none

Off class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command.Off(*args,  
**kwargs)
```

A class for SDP Subarray's Off() command.

do()

Invokes Off command on the SDP Subarray.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

off_cmd_ended_cb (*event*)

Callback function executes when the command invoked asynchronously returns from the server.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

AssignResources class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.assign_resources_command.AssignResources(*args,  
**kwargs)
```

A class for SdpSubarrayLeafNode's AssignResources() command.

AssignResources_ended (*event*)

This is the callback method of AssignResources command of the SDP Subarray. It checks whether the AssignResources command on SDP subarray is successful.

Parameters **argin** – event: response from SDP Subarray for the invoked assign resource command.

Returns None

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

do (argin)

Assigns resources to given SDP subarray. This command is provided as a noop placeholder from SDP subarray. Eventually this will likely take a JSON string specifying the resource request.

Parameters **argin** – The string in JSON format. The JSON contains following values:

SBI ID and maximum length of the SBI: Mandatory JSON object consisting of

SBI ID : String

max_length: Float

Scan types: Consist of Scan type id name

scan_type: DevVarStringArray

Processing blocks: Mandatory JSON object consisting of

processing_blocks: DevVarStringArray

Example: {“id”：“sbi-mvp01-20200325-00001”,“max_length”:100.0,“scan_types”:[{“id”：“science_A”,“coordinate_system”：“ICRS”,“ra”：“02:42:40.771”,“dec”：“-00:00:47.84”,“channels”:[{“count”:744,“start”:0,“stride”:2,“freq_min”:0.35e9,“freq_max”:0.368e9,“link_map”:[[0,0],[200,1],[744,2],[944,3]]},{“count”:744,“start”:2000,“stride”:1,“freq_min”:0.36e9,“freq_max”:0.368e9,“link_map”:[[2000,4],[2200,5]]}],{“id”：“calibration_B”,“coordinate_system”：“ICRS”,“ra”：“12:29:06.699”,“dec”：“02:03:08.598”,“channels”:[{“count”:744,“start”:0,“stride”:2,“freq_min”:0.35e9,“freq_max”:0.368e9,“link_map”:[[0,0],[200,1],[744,2],[944,3]]},{“count”:744,“start”:2000,“stride”:1,“freq_min”:0.36e9,“freq_max”:0.368e9,“link_map”:[[2000,4],[2200,5]]}]},“processing_blocks”:[{“id”：“pb-mvp01-20200325-00001”,“workflow”:{“type”：“realtime”,“id”：“vis_receive”,“version”：“0.1.0”},“parameters”:{}}, {“id”：“pb-mvp01-20200325-00002”,“workflow”:{“type”：“realtime”,“id”：“test_realtime”,“version”：“0.1.0”},“parameters”:{}}, {“id”：“pb-mvp01-20200325-00003”,“workflow”:{“type”：“batch”,“id”：“ical”,“version”：“0.1.0”},“parameters”:{}}, {“id”：“pb-mvp01-20200325-00004”,“workflow”:{“type”：“batch”,“id”：“dprep”,“version”：“0.1.0”},“parameters”:{}}, {“id”：“pb-mvp01-20200325-00003”,“workflow”:{“type”：“calibration”,“id”：“calibration”}}]}

Note: Enter input without spaces

Returns None

Raises ValueError if input argument json string contains invalid value. DevFailed if the command execution is not successful.

ReleaseResources class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_resources_command.Re
```

A class for SdpSubarrayLeafNode’s ReleaseAllResources() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises Exception if command execution throws any type of exception

do ()

Releases all the resources of given SDPSubarrayLeafNode. It accepts the subarray id, releaseALL flag and receptorIDList in JSON string format.

Parameters **argin** – None.

Returns None

Raises DevFailed if the command execution is not successful.

releaseallresources_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the releaseallresources command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

Configure class for SDPSubarrayLeafNode.

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.configure_command.Configure

A class for SdpSubarrayLeafNode's Configure() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises Exception if command execution throws any type of exception

configure_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the configure command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name

- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

do (argin)

Configures the SDP Subarray device by providing the SDP PB configuration needed to execute the receive workflow

Parameters **argin** – The string in JSON format. The JSON contains following values:

Example:

```
{ "scan_type": "science_A" }
```

Returns None

Raises ValueError if input argument json string contains invalid value. KeyError if input argument json string contains invalid key. DevFailed if the command execution is not successful

Scan class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.scan_command.Scan(*args,  
**kwargs)
```

A class for SdpSubarrayLeafNode's Scan() command.

check_allowed()

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

do (argin)

Invoke Scan command to SDP subarray.

Parameters **argin** – The string in JSON format. The JSON contains following values:

Example: {"id":1}

Note: Enter input as without spaces:{“id”:1}

Returns None

Raises DevFailed if the command execution is not successful.

scan_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the scan command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

EndScan class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.endscan_command.EndScan (*args, **kwargs)
```

A class for SdpSubarrayLeafNode's EndScan() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state**Return type** boolean**Raises** Exception if command execution throws any type of exception.**do()**

It invokes EndScan command on SdpSubarray. This command is allowed when SdpSubarray is in SCANNING state.

Parameters **argin** – None**Returns** None**Raises** DevFailed if the command execution is not successful.**endscan_cmd_ended_cb(event)**

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the endscan command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object**It has the following members:**

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise

- errors : (sequence<DevError>) The error stack
- ext

Returns none

End class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.end_command.End(*args,  
**kwargs)
```

A class for SdpSubarrayLeafNode's End() command.

```
check_allowed()
```

Checks whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises Exception if command execution throws any type of exception.

```
do()
```

This command invokes End command on SDP subarray to end the current Scheduling block.

Returns None

Raises DevFailed if the command execution is not successful.

```
end_cmd_ended_cb(event)
```

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the end command has been successfully invoked on SDP Subarray.

Parameters event – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

Abort class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.abort_command.Abort(*args,  
**kwargs)
```

A class for sdpSubarrayLeafNode's Abort() command.

```
abort_cmd_ended_cb(event)
```

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the abort command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to abort the current operation being done on the SDP subarray.

Returns None

Raises DevFailed if error occurs while invoking command on CSPSubarray.

Restart class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_command.Restart (*args, **kwargs)
```

A class for sdpSubarrayLeafNode's Restart() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to restart the SDP subarray and bring it to its ON state.

Returns None

Raises DevFailed if error occurs while invoking command on SDPSubarray.

restart_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the restart command has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

ObsReset class for SDPSubarrayLeafNode.

```
class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset_command.ObsReset (*
```

A class for SdpSubarrayLeafNode's ObsResetCommand() command.

check_allowed()

Checks whether this command is allowed to be run in current device state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do()

Command to reset the SDP subarray and bring it to its RESETTING state.

Parameters **argin** – None

Returns None

Raises DevFailed if error occurs while invoking command on SDPSubarray.

obsreset_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the ObsResetCommand has been successfully invoked on SDP Subarray.

Parameters **event** – A CmdDoneEvent object.

This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object

It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout

- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

CHAPTER 10

CSP Subarray Leaf Node

CHAPTER 11

SDP Master Leaf Node

The primary responsibility of the SDP Subarray Leaf node is to monitor the SDP Subarray and issue control actions during an observation. It also acts as a SDP contact point for Subarray Node for observation execution. There is one to one mapping between SDP Subarray Leaf Node and SDP subarray.

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
```

The primary responsibility of the SDP Subarray Leaf node is to monitor the SDP Subarray and issue control actions during an observation.

Disable()

Sets the OperatingState to Disable.

Parameters **argin** – None

Returns None

class InitCommand(*args, **kwargs)

A class for the SDP master's init_device() method”

do()

Initializes the attributes and properties of the SdpMasterLeafNode.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ReturnCode, str)

Raises

ProcessingBlockList

Used by autodoc_mock_imports.

SdpMasterFQDN

Used by autodoc_mock_imports.

Standby()

Invokes Standby command .

Parameters `argin` – None

Returns None

activityMessage
Used by autodoc_mock_imports.

always_executed_hook()
Internal construct of TANGO.

delete_device()
Internal construct of TANGO.

init_command_objects()
Initialises the command handlers for commands supported by this device.

is_Disable_allowed()
Checks Whether this command is allowed to be run in current device state.

Returns True if this command is allowed to be run in current device state.

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state.

read_ProcessingBlockList()
Internal construct of TANGO. :return:

read_activityMessage()
Internal construct of TANGO. String providing information about the current activity in SDPLeafNode.

read_versionInfo()
Internal construct of TANGO. Version information of TANGO device.

sdpHealthState
Used by autodoc_mock_imports.

versionInfo
Used by autodoc_mock_imports.

write_activityMessage (value)
Internal construct of TANGO. Sets the activity message.

`tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.main(args=None, **kwargs)`

class `tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.On(*args, **kwargs)`
A class for SDP master's On() command.

do()
Informs the SDP that it can start executing Processing Blocks. Sets the OperatingState to ON.

Parameters `argin` – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

on_cmd_ended_cb (event)
Callback function immediately executed when the asynchronous invoked command returns. Checks whether the On command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.Off(*args,  
**kwargs)
```

A class for SDP master's Off() command.

do()

Sets the OperatingState to Off.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

off_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the OFF command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.Standby(*args,  
**kwargs)
```

A class for SDP Master's Standby() command.

check_allowed()

Check Whether this command is allowed to be run in current device state.

return True if this command is allowed to be run in current device state.

rtype boolean

raises DevFailed if this command is not allowed to be run in current device state.

do()

Informs the SDP to stop any executing Processing. To get into the STANDBY state all running PBs will be aborted. In normal operation we expect diable should be triggered without first going into STANDBY.

Parameters **argin** – None.

Returns None

standby_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the standby command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.**Disable**(*args, **kwargs)

A class for SDP master's Disable() command.

check_allowed()

Check Whether this command is allowed to be run in current device state.

return True if this command is allowed to be run in current device state.

rtype boolean

raises DevFailed if this command is not allowed to be run in current device state.

disable_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the disable command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise

- errors : (sequence<DevError>) The error stack
- ext

Returns none

do ()

Sets the OperatingState to Disable.

Parameters **argin** – None.

Returns None

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.disable_command.Disable(*args,
**kwargs)
```

A class for SDP master's Disable() command.

check_allowed()

Check Whether this command is allowed to be run in current device state.

return True if this command is allowed to be run in current device state.

rtype boolean

raises DevFailed if this command is not allowed to be run in current device state.

disable_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the disable command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

do ()

Sets the OperatingState to Disable.

Parameters **argin** – None.

Returns None

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.off_command.Off(*args,
**kwargs)
```

A class for SDP master's Off() command.

do ()

Sets the OperatingState to Off.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

off_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the OFF command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.on_command.On(*args,  
**kwargs)
```

A class for SDP master's On() command.

do()

Informs the SDP that it can start executing Processing Blocks. Sets the OperatingState to ON.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (ResultCode, str)

on_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the On command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.on_command.On(*args,  
**kwargs)
```

A class for SDP master's On() command.

do ()

Informs the SDP that it can start executing Processing Blocks. Sets the OperatingState to ON.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status.

The message is for information purpose only.

Return type (resultCode, str)

on_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the On command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.standby_command.Standby(*args,  
**kwargs)
```

A class for SDP Master's Standby() command.

check_allowed ()

Check Whether this command is allowed to be run in current device state.

return True if this command is allowed to be run in current device state.

rtype boolean

raises DevFailed if this command is not allowed to be run in current device state.

do ()

Informs the SDP to stop any executing Processing. To get into the STANDBY state all running PBs will be aborted. In normal operation we expect diable should be triggered without first going into STANDBY.

Parameters **argin** – None.

Returns None

standby_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns. Checks whether the standby command has been successfully invoked on SDP Master.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

CHAPTER 12

MCCS Master Leaf Node

```
class tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMaster
```

Properties:

- MccsMasterFQDN - Property to provide FQDN of MCCS Master Device

Attributes:

- mccsHealthState - Forwarded attribute to provide MCCS Master Health State
- activityMessage - Attribute to provide activity message

AssignResources (argin)

Invokes AssignResources command on Mcccs Master

class InitCommand (*args, **kwargs)

A class for the TMC MCCS Master Leaf Node's init_device() method.

do ()

Initializes the attributes and properties of the MccsMasterLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

Raises DevFailed if error occurs while creating the device proxy for Mccs Master or subscribing the evennts.

MccsMasterFQDN

Used by autodoc_mock_imports.

ReleaseResources (argin)

Invokes ReleaseResources command on MccsMasterLeafNode

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()

Internal construct of TANGO.

delete_device()

Internal construct of TANGO.

init_command_objects()

Initialises the command handlers for commands supported by this device.

is_AssignResources_allowed()

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

is_ReleaseResources_allowed()

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

mccsHealthState

Used by autodoc_mock_imports.

read_activityMessage()

write_activityMessage(value)

tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.**main(args=None, **kwargs)**

Runs the MccsMasterLeafNode.

Parameters

- **args** – Arguments internal to TANGO
- **kwargs** – Arguments internal to TANGO

Returns An object of CompletedProcess class returned by the subprocess.

class tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.**AssignRe**

A class for MccsMasterLeafNode's AssignResources() command.

allocate_ended(event)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters event – a CmdDoneEvent object. This class is used to pass data to the call-back method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

Raises DevFailed if this command is not allowed to be run

in current device state

`check_allowed()`

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

`do (argin)`

It accepts stationIDList list, channels and stationBeamIDList in JSON string format and invokes allocate command on MccsMaster with JSON string as an input argument.

:param argin:StringType. The string in JSON format.

Example: {

```
    "subarray_id": 1, "station_ids": [1,2], "channels": [1,2,3,4,5,6,7,8], "station_beam_ids": [1]
}
```

Note: Enter the json string without spaces as an input.

Returns None

Raises ValueError if input argument json string contains invalid value KeyError if input argument json string contains invalid key DevFailed if the command execution is not successful

`class tmccprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node`.**ReleaseResources**

A class for MccsMasterLeafNode's ReleaseResources() command.

`check_allowed()`

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises ValueError if input argument json string contains invalid value DevFailed if this command is not allowed to be run in current device state

`do (argin)`

It invokes ReleaseResources command on MccsMaster and releases all the resources assigned to MccsMaster.

:param argin:StringType. The string in JSON format.

Example:

```
{ "subarray_id": 1, "release_all": true,
}
```

Returns None.

Raises DevFailed if the command execution is not successful

`releaseresources_cmd_ended_cb (event)`

Callback function immediately executed when the asynchronous invoked command returns.

Parameters `event` – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.On(*args,  
**kwargs)
```

A class for MccsMasterLeafNode's On() command.

do()

Invokes On command on the MCCS Element.

Parameters **argin** – None

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

on_cmd_ended_cb(event)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.Off(*args,  
**kwargs)
```

A class for MccsMasterLeafNode's Off() command.

do()

Invokes Off command on the MCCS Master.

Parameters **argin** – None.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ResultCode, str)

off_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

CHAPTER 13

MCCS Subarray Leaf Node

MCCS Subarray Leaf node monitors the MCCS Subarray and issues control actions during an observation. It also acts as a MCCS contact point for Subarray Node for observation execution for TMC.

class tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.**M**

MCCS Subarray Leaf node monitors the MCCS Subarray and issues control actions during an observation.

Configure (*argin*)

Invokes Configure command on MccsSubarrayLeafNode

End()

Invokes End command on MccsSubarrayLeafNode.

EndScan()

Invokes EndScan command on MccsSubarray.

class InitCommand (**args*, ***kargs*)

A class for the MccsSubarrayLeafNode's init_device() method"

do()

Initializes the attributes and properties of the MccsSubarrayLeafNode.

Returns A tuple containing a return code and a string message indicating status. The message is

for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if error occurs in creating proxy for MCCSSubarray.

MccsSubarrayFQDN

Used by autodoc_mock_imports.

Scan (*argin*)

Invokes Scan command on mccssubarrayleafnode

activityMessage

Used by autodoc_mock_imports.

always_executed_hook()
Internal construct of TANGO.

delete_device()
Internal construct of TANGO.

init_command_objects()
Initialises the command handlers for commands supported by this device.

is_Configure_allowed()
Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_EndScan_allowed()
Checks whether the command is allowed to be run in the current state.

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_End_allowed()
Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

is_Scan_allowed()
Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

mccsSubarrayHealthState
Used by autodoc_mock_imports.

mccsSubarrayObsState
Used by autodoc_mock_imports.

read_activityMessage()

write_activityMessage (*value*)

tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.**main** (*args...
**kwargs*)

class tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.**Config**

A class for MccsSubarrayLeafNode's Configure() command.

check_allowed()

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

configure_cmd_ended_cb (*event*)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- **device** : (DeviceProxy) The DeviceProxy object on which the call was executed.
- **cmd_name** : (str) The command name
- **argout_raw** : (DeviceData) The command argout
- **argout** : The command argout
- **err** : (bool) A boolean flag set to true if the command failed. False otherwise
- **errors** : (sequence<DevError>) The error stack
- **ext**

Returns none

create_cmd_data (*configuration_string*)

do (*argin*)

This command configures a scan. It accepts configuration information in JSON string format and invokes Configure command on MccsSubarray.

Parameters The string in JSON format. The JSON contains following values (*argin:DevString*) –

Example: {“stations”:[{“station_id”:1},{“station_id”:2}]},”station_beam_pointings”:[{“station_beam_id”:1,”target”:{“system_id”:1}}]}

Note: Enter the json string without spaces as a input.

Returns A tuple containing a return code and a string message indicating status. The message is for information purpose only.

Return type (ReturnCode, str)

Raises DevFailed if the command execution is not successful ValueError if input argument json string contains invalid value KeyError if input argument json string contains invalid key

get_sky_coordinates (*station_beam_pointings*)

get_station_ids (*configuration_string*)

```
update_configuration_json(station_beam_pointings, configuration_string)
```

```
class tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.S
```

A class for MccsSubarrayLeafNode's Scan() command.

```
check_allowed()
```

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

```
do(argin)
```

This command invokes Scan command on MccsSubarray. It is allowed only when MccsSubarray is in ObsState READY.

Parameters **argin** – JSON string consists of scan id (int).

Example: {"id":1}

Note: Enter the json string without spaces as a input.

Returns None

Return type Void

Raises DevFailed if the command execution is not successful

```
scan_cmd_ended_cb(event)
```

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

```
class tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.E
```

A class for MccsSubarrayLeafNode's EndScan() command.

```
check_allowed()
```

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do ()

This command invokes EndScan command on MccsSubarray. It is allowed only when MccsSubarray is in ObsState SCANNING.

Raises DevFailed if the command execution is not successful. AssertionError if MccsSubarray is not in SCANNING obsState.

endscan_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout
- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

class tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.End

A class for MccsSubarrayLeafNode's End() command.

check_allowed()

Checks whether the command is allowed to be run in the current state

Returns True if this command is allowed to be run in current device state

Return type boolean

Raises DevFailed if this command is not allowed to be run in current device state

do ()

This command invokes End command on MCCS Subarray in order to end current scheduling block.

Returns None

Return type Void

Raises DevFailed if the command execution is not successful

end_cmd_ended_cb (event)

Callback function immediately executed when the asynchronous invoked command returns.

Parameters **event** – a CmdDoneEvent object. This class is used to pass data to the callback method in asynchronous callback model for command execution.

Type CmdDoneEvent object It has the following members:

- device : (DeviceProxy) The DeviceProxy object on which the call was executed.
- cmd_name : (str) The command name
- argout_raw : (DeviceData) The command argout
- argout : The command argout

- err : (bool) A boolean flag set to true if the command failed. False otherwise
- errors : (sequence<DevError>) The error stack
- ext

Returns none

CHAPTER 14

Indices and tables

- genindex
- modindex
- search

Python Module Index

t
 tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode
 tmcprototype.centralnode.src.centralnode.assign_resources_command,
 4 75
 tmcprototype.centralnode.src.centralnode.central_node,
 1 81
 tmcprototype.centralnode.src.centralnode.release_resources_command,
 5 68
 tmcprototype.centralnode.src.centralnode.stow_antennas_command,
 6 63
 tmcprototype.centralnodelow.src.centralnodelow.assign_resources_command,
 12 65
 tmcprototype.centralnodelow.src.centralnodelow.central_node_low,
 7 68
 tmcprototype.centralnodelow.src.centralnodelow.health_state_aggregator,
 13 67
 tmcprototype.centralnodelow.src.centralnodelow.release_resources_command,
 13 70
 tmcprototype.centralnodelow.src.centralnodelow.standby_telescope_command,
 14 63
 tmcprototype.centralnodelow.src.centralnodelow.startup_telescope_command,
 15 62
 tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node,
 43 64
 tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_command,
 45 69
 tmcprototype.cspmasterleafnode.src.cspmasterleafnode.on_command,
 46 66
 tmcprototype.cspmasterleafnode.src.cspmasterleafnode.standby_command,
 46 51
 tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node,
 33 23
 tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node,
 83 20
 tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.disable_command,
 79 22
 tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.off_command,
 23 23
 tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.on_command,
 80 22
 24
 tmcprototype.subarraynode.src.subarraynode.abort_command
 tmcprototype.subarraynode.src.subarraynode.assign_resources_command
 tmcprototype.subarraynode.src.subarraynode.config_command
 tmcprototype.subarraynode.src.subarraynode.end_command
 tmcprototype.subarraynode.src.subarraynode.end_scanning_command
 tmcprototype.subarraynode.src.subarraynode.obsreset_command

```
tmcprototype.subarraynode.src.subarraynode.off_command,  
    19  
tmcprototype.subarraynode.src.subarraynode.on_command,  
    19  
tmcprototype.subarraynode.src.subarraynode.release_all_resources_command,  
    21  
tmcprototype.subarraynode.src.subarraynode.restart_command,  
    24  
tmcprototype.subarraynode.src.subarraynode.scan_command,  
    22  
tmcprototype.subarraynode.src.subarraynode.subarray_node,  
    17  
tmcprototype.subarraynode.src.subarraynode.track_command,  
    24  
tmcprototype.subarraynodelow.src.subarraynodelow.assign_resources_command,  
    29  
tmcprototype.subarraynodelow.src.subarraynodelow.configure_command,  
    29  
tmcprototype.subarraynodelow.src.subarraynodelow.end_command,  
    29  
tmcprototype.subarraynodelow.src.subarraynodelow.end_scan_command,  
    30  
tmcprototype.subarraynodelow.src.subarraynodelow.off_command,  
    30  
tmcprototype.subarraynodelow.src.subarraynodelow.on_command,  
    30  
tmcprototype.subarraynodelow.src.subarraynodelow.release_all_resources_command,  
    30  
tmcprototype.subarraynodelow.src.subarraynodelow.scan_command,  
    31  
tmcprototype.subarraynodelow.src.subarraynodelow.subarray_node_low,  
    27
```

Index

A

Abort (class in *tmcproto-* type.*mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarrayMessage* attribute), 89
Abort (class in *tmcproto-* type.*sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node* attribute), 76
Abort (class in *tmcproto-* type.*sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node* attribute), 76
Abort () (*tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node* attribute), 52
Abort () (*tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node* attribute), 57
Abort () (*tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node* attribute), 57
abort_cmd_ended_cb () (*tmcproto-* activityMessage type.*sdpsubarrayleafnode.abort_command* attribute), 27
abort_cmd_ended_cb () (*tmcproto-* add_receptors_in_group () type.*sdpsubarrayleafnode.sdp_subarray_leaf_node*.*Abort* attribute), 20
abortCommand (class in *tmcproto-* allocate_ended () type.*mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node* attribute), 84
activeProcessingBlocks (*tmcproto-* always_executed_hook () type.*sdpsubarrayleafnode.sdp_subarray_leaf_node*.*CentralNode* attribute), 2
activityMessage (*tmcproto-* always_executed_hook () type.*centralnode.src.centralnode.central_node*.*CentralNode* attribute), 2
activityMessage (*tmcproto-* always_executed_hook () type.*centralnode.src.centralnode.central_node_low*.*CentralNode* attribute), 8
activityMessage (*tmcproto-* always_executed_hook () type.*cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node* attribute), 43
activityMessage (*tmcproto-* always_executed_hook () type.*cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node* attribute), 43
activityMessage (*tmcproto-* always_executed_hook () type.*cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node* attribute), 83
activityMessage (*tmcproto-* always_executed_hook () type.*dishleafnode.src.dishleafnode.dish_leaf_node*.*DishLeafNode* attribute), 89
activityMessage (*tmcproto-* always_executed_hook () type.*mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node* attribute), 83
activityMessage (*tmcproto-* always_executed_hook () type.*sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node* attribute), 76
activityMessage (*tmcproto-* always_executed_hook () type.*sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node* attribute), 76

```

        method), 52
always_executed_hook()          (tmproto- AssignResources()           (tmproto-
    type.subarraynode.src.subarraynode.subarray_node.SubarrayNodeCentralnodeLow.src.centralnodeLow.central_node_low.Central
    method), 18                         method), 7
                                         method), 7
always_executed_hook()          (tmproto- AssignResources()           (tmproto-
    type.subarraynodeLow.src.subarraynodeLow.subarray_node_type_SubarrayNodeLeafnode.src.mccsmasterleafnode.mccs_master_l
    method), 27                         method), 83
                                         method), 83
assign_csp_resources()          (tmproto- AssignResources()           (tmproto-
    type.subarraynode.src.subarraynode.assign_resources_command.type_SubarrayLeafnode.sdp_subarray
    method), 20                         method), 51
                                         method), 51
assign_resources_leaf_node()    (tmproto- AssignResources_ended()      (tmproto-
    type.centralnodeLow.src.centralnodeLow.assign_resources_command.type_SubarrayLeafnode.sdp_subarrayleafnode.assign_resour
    method), 12                         method), 63
                                         method), 63
assign_resources_leaf_node()    (tmproto- AssignResources_ended()      (tmproto-
    type.centralnodeLow.src.centralnodeLow.central_node_low.AssignResourcesSubarrayLeafnode.sdp_subarrayleafnode.sdp_subarray
    method), 9                           method), 54
                                         method), 54
assign_resources_mccs()         (tmproto- AssignResourcesCommand (class in tmproto-
    type.centralnodeLow.src.centralnodeLow.assign_resources_command.type_SubarrayLeafnode.assign_resources_command
    method), 12                         20
                                         20
assign_resources_mccs()         (tmproto- AssignResourcesCommand (class in tmproto-
    type.centralnodeLow.src.centralnodeLow.central_node_low.AssignResourcesSubarrayLeafnode.sdp_subarrayassign_resources_com
    method), 9                           29
                                         29
assign_resources_subarray()     (tmproto- attribute_event_handler()      (tmproto-
    type.centralnodeLow.src.centralnodeLow.assign_resources_command.type_SubarrayLeafnode.dish_leaf_node.DishLeafNode
    method), 12                         method), 38
                                         method), 38
assign_resources_subarray()     (tmproto- B
    type.centralnodeLow.src.centralnodeLow.central_node_low.AssignResources
    method), 9                         build_up_csp_cmd_data()       (tmproto-
                                         type.subarraynode.src.subarraynode.configure_command.Element
                                         type.subarraynode.src.subarraynode.assign_resources_command
    method), 20                         build_up_dsh_cmd_data()       (tmproto-
                                         type.subarraynode.src.subarraynode.configure_command.Element
                                         type.subarraynode.src.subarraynode.assign_resources_command
    method), 4                           build_up_sdp_cmd_data()       (tmproto-
                                         type.subarraynode.src.subarraynode.configure_command.Element
                                         type.subarraynode.src.subarraynode.assign_resources_command
    method), 22                         22
                                         22
AssignResources (class in tmproto- C
    type.centralnodeLow.src.centralnodeLow.central_node_low.calculate_observation_state() (tmproto-
    9                                         type.subarraynode.src.subarraynode.subarray_node.SubarrayNo
                                         type.subarraynode.src.subarraynode.calculate_observation_state() (tmproto-
                                         type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node)
                                         method), 18
                                         method), 18
                                         type.subarraynodeLow.src.subarraynodeLow.subarray_node_low.S
                                         type.subarraynodeLow.src.subarraynodeLow.assign_resources_command)
                                         method), 27
                                         method), 27
                                         type.subarraynodeLow.src.subarraynodeLow.call_end_scan_command() (tmproto-
                                         type.subarraynodeLow.src.subarraynodeLow.sdp_subarrayleafnode)
                                         63                                         call_end_scan_command() (tmproto-
                                         type.subarraynodeLow.src.subarraynodeLow.sdp_subarrayleafnode)
                                         54                                         call_end_scan_command() (tmproto-
                                         type.subarraynodeLow.src.subarraynodeLow.sdp_subarrayleafnode)
                                         54                                         call_end_scan_command() (tmproto-
                                         type.subarraynodeLow.src.subarraynodeLow.scan_command.ScanComma
                                         type.subarraynodeLow.src.subarraynodeLow.scan_command)
                                         method), 31
                                         method), 31
                                         type.subarraynodeLow.src.subarraynodeLow.scan_command.ScanComma
                                         type.subarraynodeLow.src.subarraynodeLow.scan_command)
                                         3                                         CentralAlarmHandler (tmproto-
                                         type.centralnode.src.centralnode.central_node)
                                         3                                         CentralAlarmHandler (tmproto-
                                         type.centralnode.src.centralnode.central_node)
                                         type.centralnode.src.centralnode.central_node.CentralNode
                                         type.centralnode.src.centralnode.central_node)
                                         attribute), 1
                                         attribute), 1

```

CentralAlarmHandler (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.CentralNodeLow_src.dishleafnode.dish_leaf_node.DishLeafNode.
 attribute), 7 method), 33

CentralNode (class in tmproto- check_allowed() (tmproto-
 type.centralnode.src.centralnode.central_node), type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.
 1 method), 33

CentralNode (class in tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.
 7 method), 34

CentralNode.InitCommand (class in tmproto- check_allowed() (tmproto-
 type.centralnode.src.centralnode.central_node), type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.
 1 method), 34

CentralNode.InitCommand (class in tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.
 7 method), 35

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode.src.centralnode.assign_resources_command.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 4 method), 35

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode.src.centralnode.release_resources_command.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 5 method), 36

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode.src.centralnode.stow_antennas_command.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 6 method), 36

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 12 method), 37

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 9 method), 37

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.ReleaseDishleafnode.type.dishleafnode.dish_leaf_node.DishLeafNode.
 method), 10 method), 37

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.StopByTelescope.typeleafnode.src.mccsmasterleafnode.mccs_master_l.
 method), 10 method), 85

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.StopUpDateSubarrayleafnode.src.mccsmasterleafnode.mccs_master_l.
 method), 11 method), 85

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.ReleaseResourcesByFlexfunction.typeleafnode.src.mccssubarrayleafnode.mccs_suba.
 method), 13 method), 91

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.StandbyTelescope.typeleafnode.src.mccssubarrayleafnode.mccs_suba.
 method), 14 method), 93

check_allowed() (tmproto- check_allowed() (tmproto-
 type.centralnode_low.src.centralnode_low.StartupTelescope.typeleafnode.src.mccssubarrayleafnode.mccs_suba.
 method), 15 method), 92

check_allowed() (tmproto- check_allowed() (tmproto-
 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.typeleafnode.src.mccssubarrayleafnode.mccs_suba.
 method), 45 method), 92

check_allowed() (tmproto- check_allowed() (tmproto-
 type.cspmasterleafnode.src.cspmasterleafnode.standby_command.typeleafnode.src.sdpmasterleafnode.disable_command.
 method), 46 method), 79

```

check_allowed()          (tmproto- check_allowed()          (tmproto-
    type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_type_standaloneleafnode.src.sdpsubarrayleafnode.sdp_subarray_
    method), 78           method), 55
check_allowed()          (tmproto- check_allowed()          (tmproto-
    type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_type_standaloneleafnode.src.sdpsubarrayleafnode.sdp_subarray_
    method), 77           method), 58
check_allowed()          (tmproto- check_allowed()          (tmproto-
    type.sdpmasterleafnode.src.sdpmasterleafnode.standby_contype_standbyleafnode.src.sdpsubarrayleafnode.sdp_subarray_
    method), 81           method), 60
check_allowed()          (tmproto- check_allowed()          (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.abort_cause_and_abortnode.src.subarraynode.track_command.TrackComm_
    method), 69           method), 24
check_allowed()          (tmproto- command_class_object() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.assign_resource_subarraynode.AssignResourceSubarray_node.SubarrayNo_
    method), 63           method), 18
check_allowed()          (tmproto- command_class_object() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.configure_type_and_configuredare_src.subarraynode_low.subarray_node_low.S_
    method), 65           method), 27
check_allowed()          (tmproto- Configure      (class      in      tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.end_command_endsubarrayleafnode.src.mcssubarrayleafnode.mccs_suba_
    method), 68           91
check_allowed()          (tmproto- Configure      (class      in      tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.endscan_type_and_endsubarrayleafnode_src_sdpsubarrayleafnode.configure_com_
    method), 67           65
check_allowed()          (tmproto- Configure      (class      in      tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset_type_and_obsresetnode_src_sdpsubarrayleafnode.sdp_subarray_
    method), 70           57
check_allowed()          (tmproto- Configure      (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_repeatedleafnode_and_ReleaseAllResidhcdaf_node.DishLeafNode_
    method), 64           method), 33
check_allowed()          (tmproto- Configure      (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_dynamickRestartleafnode_src_mcssubarrayleafnode.mccs_suba_
    method), 69           method), 89
check_allowed()          (tmproto- Configure      (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.scan_commandScanSubarrayleafnode_src_sdpsubarrayleafnode.sdp_subarray_
    method), 66           method), 51
check_allowed()          (tmproto- configure_cmdEnded_cb() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_src_mcssubarrayleafnode.mccs_suba_
    method), 58           method), 91
check_allowed()          (tmproto- configure_cmdEnded_cb() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_ConfiguredEndsSubarrayleafnode.configure_com_
    method), 54           method), 65
check_allowed()          (tmproto- configure_cmdEnded_cb() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_ConfiguredEndsSubarrayleafnode.sdp_subarray_
    method), 57           method), 57
check_allowed()          (tmproto- ConfigureCommand (class      in      tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_ConfiguredEndsSubarraynode.configure_command),
    method), 60           22
check_allowed()          (tmproto- ConfigureCommand (class      in      tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_ConfiguredEndsSubarraynode_low.configure_command)
    method), 61           29
check_allowed()          (tmproto- convert_radec_to_azel() (tmproto-
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subtype_pays_dish_leafnode_ConfiguredEndsSubarraynode_low.dish_leaf_node.DishLeafNode_
    method), 59           method), 38

```

```

create_cmd_data ()          (tmcproto- DeviceData      (in      module      tmcproto-
    type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_type_subarrayleafnode_configurable.central_node),
    method), 91               3
cspHealthState              (tmcproto- Disable       (class      in      tmcproto-
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode.sdpmasterleafnode.disable_command
    attribute), 43             79
CspMasterFQDN                (tmcproto- Disable       (class      in      tmcproto-
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode.sdpmasterleafnode.sdp_master_leaf_node
    attribute), 43             78
CspMasterLeafNode   (class      in      tmcproto- Disable()           (tmcproto-
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode.sdpmasterleafnode.sdp_master_leaf_node
    43                         method), 75
CspMasterLeafNode.InitCommand (class      in      tmcproto- disable_cmdEnded_cb()   (tmcproto-
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode.sdpmasterleafnode.sdp_master_leaf_node
    43                         method), 79
CspMasterLeafNode.FQDN        (tmcproto- disable_cmdEnded_cb()   (tmcproto-
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode.sdpmasterleafnode.sdp_master_leaf_node
    43                         method), 79
CspMasterLeafNodeFQDN        (tmcproto- type.sdpmasterleafnode.sdp_master_leaf_node.DishLeafNode
    type.centralnode.src.centralnode.central_node.CentralNode.method), 78
    attribute), 1               dishHealthState         (tmcproto-
                                DishLeafNode      (class      in      tmcproto-
    type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 38
    attribute), 17              type.dishleafnode.sdp_master_leaf_node.DishLeafNode
                                DishLeafNode      (class      in      tmcproto-
    type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
    attribute), 17              type.dishleafnode.sdp_master_leaf_node.DishLeafNode
                                DishLeafNode.AbstCommand (class in tmcproto-
    type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
    33                           type.dishleafnode.sdp_master_leaf_node.DishLeafNode),
D
delete_device()              (tmcproto- DishLeafNode.ConfigureCommand
    type.centralnode.src.centralnode.central_node.CentralNode(class      in      tmcproto-
    method), 2                   type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode
    type.dishleafnode.sdp_master_leaf_node.DishLeafNode),
    33
delete_device()              (tmcproto- DishLeafNode.CleanAndScanCommand
    type.centralnode.src.centralnode.central_node.CentralNode(class      in      tmcproto-
    method), 8                   type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode.InitCommand (class in tmcproto-
    type.dishleafnode.sdp_master_leaf_node.DishLeafNode),
    34
delete_device()              (tmcproto- DishLeafNode.DishLeafNode.CspMasterLeafNode
    type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode(class      in      tmcproto-
    method), 44                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode)
    34
delete_device()              (tmcproto- DishLeafNode.DishLeafNode.MccsMasterLeafNode
    type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMasterLeafNode(class      in      tmcproto-
    method), 83                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode.ObsResetCommand
    34
delete_device()              (tmcproto- DishLeafNode.DishLeafNode.MccsMasterLeafNode
    type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarrayleafnode.MccsSubarrayLeafNode(class      in      tmcproto-
    method), 90                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode),
    34
delete_device()              (tmcproto- DishLeafNode.RestartCommand
    type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode(class      in      tmcproto-
    method), 76                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode),
    35
delete_device()              (tmcproto- DishLeafNode.SdpMasterLeafNode.SdpSubarrayLeafNode
    type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.SdpSubarrayLeafNode(class      in      tmcproto-
    method), 52                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode),
    35
delete_device()              (tmcproto- DishLeafNode.SdpSubarrayLeafNode.SdpSubarrayLeafNode
    type.subarraynode.src.subarraynode.subarray_node.SubarrayNode(class      in      tmcproto-
    method), 18                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode),
    35
delete_device()              (tmcproto- DishLeafNode.SdpSubarrayLeafNode.SdpSubarrayLeafNode
    type.subarraynode.src.subarraynode.subarray_node.SubarrayNode(class      in      tmcproto-
    method), 27                  type.dishleafnode.sdp_master_leaf_node.DishLeafNode,
                                DishLeafNode),
    35

```

```

(class           in      tmcproto-      method), 7
type.dishleafnode.src.dishleafnode.dish_leaf_node) do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.
35                                         method), 10
DishLeafNode .SetStandbyLPModeCommand      do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.
(class           in      tmcproto-      method), 10
type.dishleafnode.src.dishleafnode.dish_leaf_node) do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.
36                                         method), 11
DishLeafNode .SetStowModeCommand          do () (tmcprototype.centralnode_low.src.centralnode_low.release_resources_.
(class           in      tmcproto-      method), 14
type.dishleafnode.src.dishleafnode.dish_leaf_node) do () (tmcprototype.centralnode_low.src.centralnode_low.standby_telescope_.
36                                         method), 14
DishLeafNode .SlewCommand (class in tmcproto- do () (tmcprototype.centralnode_low.src.centralnode_low.startup_telescope_.
type.dishleafnode.src.dishleafnode.dish_leaf_node),      method), 15
36                                         do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_.
DishLeafNode .StartCaptureCommand        do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_.
(class           in      tmcproto-      method), 43
type.dishleafnode.src.dishleafnode.dish_leaf_node),      method), 45
36                                         do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_.
DishLeafNode .StopCaptureCommand         do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_.
(class           in      tmcproto-      method), 44
type.dishleafnode.src.dishleafnode.dish_leaf_node),      method), 45
37                                         do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_command_.
DishLeafNode .StopTrackCommand          do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.on_command_.
(class           in      tmcproto-      method), 46
type.dishleafnode.src.dishleafnode.dish_leaf_node),      method), 46
37                                         do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.standby_com-
DishLeafNode .TrackCommand (class in tmcproto- method), 46
type.dishleafnode.src.dishleafnode.dish_leaf_node) do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
37                                         method), 33
DishLeafNodePrefix                  (tmcproto- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
type.centralnode.src.centralnode.central_node.CentralNodeMethod), 33
attribute), 1                      do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
DishLeafNodePrefix                  (tmcproto- method), 34
type.subarraynode.src.subarraynode.subarray_node.SubarrayNodeMethod), 34
attribute), 17                     do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
DishMasterFQDN                    (tmcproto- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafMethod), 34
attribute), 34                     do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
dishPointingState                (tmcproto- method), 35
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafMethod), 35
attribute), 38                     do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
do () (tmcprototype.centralnode.src.centralnode.assign_resources(assignResources), 35
method), 4                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 35
do () (tmcprototype.centralnode.src.centralnode.central_node.CentralNodeTypeDishleafnodeMethod), 35
method), 1                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 35
do () (tmcprototype.centralnode.src.centralnode.release_resources(releaseResources), 36
method), 5                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 36
do () (tmcprototype.centralnode.src.centralnode.stow_antennas(stowAntennas), 36
method), 6                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 36
do () (tmcprototype.centralnode_low.src.centralnode_low.assign_resources(assignResources), 36
method), 12                          do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 36
do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_typeAssignResources), 36
method), 9                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 36
do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_typeReleaseResources), 36
method), 9                           do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLea-
method), 36

```

```

        method), 37
do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DishLeafNode::SporadicClearnode).src.sdpsubarrayleafnode.off_command
        method), 37
do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DishLeafNode::SpurkClearnode).src.sdpsubarrayleafnode.on_command
        method), 37
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.release_
        method), 85
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.MccMasterLeaveAndClearnode.restart_
        method), 83
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.MccMasterLeaveAndClearnode.restart_
        method), 69
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.Offleafnode.src.sdpsubarrayleafnode.scan_command
        method), 86
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.sdp_sub_
        method), 86
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.sdp_sub_
        method), 58
do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 85
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.sdp_sub_
        method), 91
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.sdp_sub_
        method), 57
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.sdpsubarrayleafnode.sdp_sub_
        method), 93
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 61
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 92
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 59
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 89
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 56
do () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.(mcprototype_beforwakeup).src.RelayAndReserve.sdp_subarrayleafnode.sdp_sub_
        method), 92
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 79
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.off_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 79
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.on_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 80, 81
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 79
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 60
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 77
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.sdpsubarrayleafnode.sdp_sub_
        method), 51
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.subarraynode.abort_command.Abort_
        method), 76
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.subarraynode.end_command.EndCommand_
        method), 23
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.subarraynode.end_command.EndCommand_
        method), 75
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.subarraynode.configure_command.Configure_
        method), 78
do () (tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode.sdp_(tmcprototype_Disable).src.subarraynode.end_command.EndCommand_
        method), 81
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.end_scan_command.EndScanCommand_
        method), 69
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.end_scan_command.EndScanCommand_
        method), 23
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.end_scan_command.EndScanCommand_
        method), 64
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.end_scan_command.EndScanCommand_
        method), 24
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.off_command.OffCommand_
        method), 66
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.on_command.OnCommand_
        method), 68
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.release_all_resources_
        method), 67
do () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.(tmcprototype_End).src.subarraynode.restart_command.RestartCommand_
        method), 21

```

```

        method), 24
do () (tmcprototype.subarraynode.src.subarraynode.scan_command_and_send_command_in_tmcproto-
        method), 23
do () (tmcprototype.subarraynode.src.subarraynode.subarray_node.SubarrayNode.InitCommand
        method), 17
do () (tmcprototype.subarraynode.src.subarraynode.track_command_type_and_subarrayleafnode.src.mccssubarrayleafnode.mccs_suba-
        method), 24
do () (tmcprototype.subarraynode_low.src.subarraynode_low.end_resources_command_and_assign_resources_ctmap_and-
        method), 29
do () (tmcprototype.subarraynode_low.src.subarraynode_low.configure_command_and_configure_command
        method), 29
do () (tmcprototype.subarraynode_low.src.subarraynode_low.end_commpads_and_dish_leafnode.src.sdpsubarrayleafnode.sdp_subarray-
        method), 29
do () (tmcprototype.subarraynode_low.src.subarraynode_low.end_scan_command_end_scan_command
        method), 30
do () (tmcprototype.subarraynode_low.src.subarraynode_low.off_command_and_off_command
        method), 30
do () (tmcprototype.subarraynode_low.src.subarraynode_low.on_command_and_on_command
        method), 30
do () (tmcprototype.subarraynode_low.src.subarraynode_low.release_all_resources_command_and_release_all_resources_command
        method), 30
do () (tmcprototype.subarraynode_low.src.subarraynode_low.scan_command_and_scan_command
        method), 31
do () (tmcprototype.subarraynode_low.src.subarraynode_low.subarray_low_update_subarray_low_and_update_low
        method), 27
        endscan_cmd_ended_cb () (tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray-
        method), 93
        endscan_cmd_ended_cb () (tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.endscan_com-
        method), 67
        endscan_cmd_ended_cb () (tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray-
        method), 61
End (class in tmcproto-
        type.mccssubarrayleafnode.src.mccssubarrayleafnode.end_scan_command_and_subarray_leaf_node(class in tmcproto-
        type.subarraynode.src.subarraynode.end_scan_command),
        93
End (class in tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.end_command_and (class in tmcproto-
        type.subarraynode_low.src.subarraynode_low.end_scan_command)
        68
End (class in tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node),
        60
End () (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode_and_subarray_leaf_node.MccsSubarrayLeafNode
        method), 89
        type.subarraynode.src.subarraynode.subarray_node.SubarrayNo
End () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.get_deviceproxy_and_sdpsubarrayleafnode.SdpSubarrayLeafNode
        method), 51
        get_deviceproxy () (tmcproto-
        type.subarraynode_low.src.subarraynode_low.subarray_node_low.Sd
end_cmd_ended_cb () (tmcproto-
        type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_member_and_subarray_leaf_node.End
        method), 93
        get_sky_coordinates () (tmcproto-
        type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
end_cmd_ended_cb () (tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.end_command_and_End
        method), 68
        get_station_ids () (tmcproto-
        type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
end_cmd_ended_cb () (tmcproto-
        type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_low_and_End
        method), 61
EndCommand (class in tmcproto-
        type.subarraynode.src.subarraynode.end_command_and_end_command)
        health_state_cb () (tmcproto-
        type.subarraynode_low.src.subarraynode_low.end_command_and_end_command)
        91

```

```

type.centralnode_low.src.centralnode_low.health_state_aggregatorHealthStateAggregator
method), 13                                         is_AssignResources_allowed()      (tmproto-
health_state_cb()                                     (tmproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 84
method), 18                                         is_AssignResources_allowed()      (tmproto-
health_state_cb()                                     (tmproto-      type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
type.subarraynode_low.src.subarraynode_low.subarray_node.healthSubarrayNode
method), 28                                         is_Configure_allowed()          (tmproto-
HealthStateAggregator (class in tmproto-      type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
type.centralnode_low.src.centralnode_low.health_state_aggregator), 38
13                                                 is_Configure_allowed()          (tmproto-
|                                         type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
init_command_objects()                               (tmproto-  is_Configure_allowed()          (tmproto-
type.centralnode.src.centralnode.central_node.CentralNode), 2
method), 2                                         is_Disable_allowed()           (tmproto-
init_command_objects()                               (tmproto-  is_End_allowed()              (tmproto-
type.centralnode_low.src.centralnode_low.CentralNode), 76
method), 8                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_EndScan_allowed()         (tmproto-
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_node.CspMasterLeafNode), 90
method), 44                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_EndScan_allowed()         (tmproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode), 52
method), 38                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_EndScan_allowed()         (tmproto-
type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_ma
method), 84                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_EndScan_allowed()         (tmproto-
type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
method), 90                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_EndScan_allowed()         (tmproto-
type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_ma
method), 76                                         is_EndScan_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_ObsReset_allowed()        (tmproto-
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_suba
method), 52                                         is_ObsReset_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_ObsReset_allowed()        (tmproto-
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 52
method), 18                                         is_ObsReset_allowed()          (tmproto-
init_command_objects()                               (tmproto-  is_ReleaseAllResources_allowed() (tmproto-
type.subarraynode_low.src.subarraynode_low.subarray_node.healthSubarrayNode), 52
method), 28                                         is_ReleaseAllResources_allowed() (tmproto-
is_Abort_allowed()                                 (tmproto-  is_ReleaseResources_allowed() (tmproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode), 2
method), 38                                         is_ReleaseResources_allowed() (tmproto-
is_Abort_allowed()                                 (tmproto-  is_ReleaseResources_allowed() (tmproto-
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_suba
method), 52                                         is_ReleaseResources_allowed() (tmproto-
is_AssignResources_allowed()                     (tmproto-  is_ReleaseResources_allowed() (tmproto-
type.centralnode.src.centralnode.central_node.CentralNode), 2
method), 2                                         is_ReleaseResources_allowed() (tmproto-
is_AssignResources_allowed()                     (tmproto-  is_Restart_allowed()          (tmproto-
type.centralnode_low.src.centralnode_low.CentralNode), 2
method), 2                                         is_Restart_allowed()          (tmproto-

```

```

        method), 38
is_Restart_allowed()           main()      (in      module      tmcproto-
                               (tmcproto-      type.centralnode_low.src.centralnode_low.central_node_low),
                               type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
                               method), 53
is_Scan_allowed()             main()      (in      module      tmcproto-
                               (tmcproto-      type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_n
                               type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
                               method), 39
is_Scan_allowed()             main()      (in      module      tmcproto-
                               (tmcproto-      type.dishleafnode.src.dishleafnode.dish_leaf_node),
                               type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_30_subarray_leaf_node.MccsSubarrayLeafNode
                               method), 90
is_Scan_allowed()             main()      (in      module      tmcproto-
                               (tmcproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
                               type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
                               method), 53
is_Slew_allowed()              main()      (in      module      tmcproto-
                               (tmcproto-      type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
                               type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
                               method), 39
is_Standby_allowed()           main()      (in      module      tmcproto-
                               (tmcproto-      type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_lea
                               type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
                               method), 44
is_StandByTelescope_allowed() main()      (in      module      tmcproto-
                               (tmcproto-      type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_
                               type.centralnode.src.centralnode.central_node.CentralNode53
                               method), 2
is_StandByTelescope_allowed() main()      (in      module      tmcproto-
                               (tmcproto-      type.subarraynode.src.subarraynode.subarray_node),
                               type.centralnode_low.src.centralnode_low.CentralNode
                               method), 8
is_StartCapture_allowed()     main()      (in      module      tmcproto-
                               (tmcproto-      type.subarraynode_low.src.subarraynode_low.subarray_node_low),
                               type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
                               method), 39
mccs_health_subscribe_event() (tmcproto-
is_StartUpTelescope_allowed() main()      (in      module      tmcproto-
                               (tmcproto-      type.centralnode_low.src.centralnode_low.health_state_aggregat
                               type.centralnode.src.centralnode.central_node.CentralNodemethod), 13
                               method), 2
mccsHealthState               (tmcproto-
is_StartUpTelescope_allowed() main()      (in      module      tmcproto-
                               (tmcproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
                               type.centralnode_low.src.centralnode_low.CentralNode)84
                               method), 8
MccsMasterFQDN                (tmcproto-
is_StopCapture_allowed()      main()      (in      module      tmcproto-
                               (tmcproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
                               type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNodeattribute), 83
                               method), 39
MccsMasterLeafNode             (class      in      tmcproto-
is_StopTrack_allowed()        main()      (in      module      tmcproto-
                               (tmcproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
                               type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
                               method), 39
MccsMasterLeafNode .InitCommand
is_StowAntennas_allowed()     main()      (class      in      tmcproto-
                               (tmcproto-      type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
                               type.centralnode.src.centralnode.central_node.CentralNode)83
                               method), 3
is_Track_allowed()             main()      (MCCSMasterLeafNodeFQDN      (tmcproto-
                               (tmcproto-      type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafN
                               type.centralnode_low.src.centralnode_low.CentralNodeattribute), 7
                               method), 39
is_Track_allowed()             main()      (tmcproto-      MccsSubarrayFQDN      (tmcproto-
                               (tmcproto-      type.subarraynode.src.subarraynode.subarray_node.SubarrayNode)89
                               type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
                               method), 18
MccsSubarrayFQDN               (tmcproto-
main()      (in      module      tmcproto-      type.subarraynode_low.src.subarraynode_low.subarray_node_low.S
                               type.centralnode.src.centralnode.central_node), 3
mccsSubarrayHealthState         mccsSubarrayHealthState      (tmcproto-
                               type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba

```

M

		attribute), 90	Off	(class in tmcproto-
MccsSubarrayLeafNode	(class in tmcproto-	type.cspmasterleafnode.src.cspmasterleafnode.off_command),		
	type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_89	subarray_leaf_node),	Off	(class in tmcproto-
MccsSubarrayLeafNode	.InitCommand			type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
	(class in tmcproto-		86	
	type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node), in tmcproto-			
	89			type.sdpmasterleafnode.src.sdpmasterleafnode.off_command),
MccsSubarrayLNFQDN	(tmcproto-		79	
	type.subarraynode_low.src.subarraynode_low.subarray_low.SubarrayNode	in tmcproto-		
	attribute), 27	type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_		
mccsSubarrayObsState	(tmcproto-		77	
	type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command		
	attribute), 90		63	
N				
NumDishes	(tmcproto-			Off (class in tmcproto-
	type.centralnode.src.centralnode.central_node.CentralNode	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray	56	
	attribute), 1	off_cmdEnded_cb ()		(tmcproto-
O				type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
observation_state_cb ()	(tmcproto-	off_cmdEnded_cb ()		method), 86
	type.subarraynode.src.subarraynode.subarray_node.SubarrayNode	type.sdpmasterleafnode.src.sdpmasterleafnode.off_command.Off		
	method), 18	method), 80		
observation_state_cb ()	(tmcproto-	off_cmdEnded_cb ()		(tmcproto-
	type.subarraynode_low.src.subarraynode_low.subarray_node_low.SubarrayNode	type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_		
	method), 28	method), 77		
ObsReset	(class in tmcproto-	off_cmdEnded_cb ()		(tmcproto-
	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command.		
	70	method), 63		
ObsReset	(class in tmcproto-	off_cmdEnded_cb ()		(tmcproto-
	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray		
	59	method), 56		
ObsReset ()	(tmcproto-	OffCommand	(class in tmcproto-	
	type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode	type.subarraynode.src.subarraynode.off_command),	19	
	method), 34			
ObsReset ()	(tmcproto-	OffCommand	(class in tmcproto-	
	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray	type.subarraynode_low.src.subarraynode_low.off_command),	30	
	method), 51			
obsreset_cmdEnded_cb ()	(tmcproto-	On	(class in tmcproto-	
	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset	type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_n	44	
	method), 70	method), ObsReset		
obsreset_cmdEnded_cb ()	(tmcproto-	On	(class in tmcproto-	
	type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray	type.cspmasterleafnode_src.cspmasterleafnode.on_command),	46	
	method), 59			
ObsResetCommand	(class in tmcproto-	On	(class in tmcproto-	
	type.subarraynode.src.subarraynode.obsreset_command),	type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l	86	
	24			
ObsStateAggregator	(in module tmcproto-	On	(class in tmcproto-	
	type.centralnode.src.centralnode.central_node),	type.sdpmasterleafnode.src.sdpmasterleafnode.on_command),	80	
	3			
Off	(class in tmcproto-	On	(class in tmcproto-	
	type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf	type.sdpmasterleafnode_src.sdpmasterleafnode.sdp_master_leaf_	76	
	45			

```

On      (class       in       tmcproto-           method), 44
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_Command) tyMessage ()          (tmcproto-
62                                         type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode

On      (class       in       tmcproto-           method), 39
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_SubarrayLeafnode) ege ()          (tmcproto-
55                                         type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l

on_cmd_ended_cb ()          (tmcproto-           method), 84
       type.cspmasterleafnode.src.cspmasterleafnode.csp_ma terleafnode) Message ()          (tmcproto-
method), 44                                         type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba

on_cmd_ended_cb ()          (tmcproto-           method), 90
       type.cspmasterleafnode.src.cspmasterleafnode.on_Command) vityMessage ()          (tmcproto-
method), 46                                         type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_ma ster_lease

on_cmd_ended_cb ()          (tmcproto-           method), 76
       type.mccsmasterleafnode.src.mccsmasterleafnode.meas_ma sterLeafnode) on_()          (tmcproto-
method), 86                                         type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray

on_cmd_ended_cb ()          (tmcproto-           method), 53
       type.sdpmasterleafnode.src.sdpmasterleafnode.on_Command) vityMessage ()          (tmcproto-
method), 80, 81                                         type.subarraynode.src.subarraynode.subarray_node.SubarrayNo

on_cmd_ended_cb ()          (tmcproto-           method), 18
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_ma sterleafnode) on_()          (tmcproto-
method), 76                                         type.subarraynode_low.src.subarraynode_low.subarray_node_low.S

on_cmd_ended_cb ()          (tmcproto-           method), 28
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_Command) eadingBlockList ()          (tmcproto-
method), 62                                         type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_ma ster_lease

on_cmd_ended_cb ()          (tmcproto-           method), 76
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_SubarrayLeafnode) es ()          (tmcproto-
method), 55                                         type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray

OnCommand (class       in       tmcproto-           method), 53
       type.subarraynode.src.subarraynode.on_Command) ead_receptorIDList ()          (tmcproto-
19                                         type.subarraynode.src.subarraynode.subarray_node.SubarrayNo

OnCommand (class       in       tmcproto-           method), 18
       type.subarraynode_low.src.subarraynode_low.on_Command) bID ()          (tmcproto-
30                                         type.subarraynode.src.subarraynode.subarray_node.SubarrayNo

P
pointing_state_cb ()          (tmcproto-           method), 18
       type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
                                         read_scanID ()          (tmcproto-
                                         type.subarraynode.src.subarraynode.subarray_node.SubarrayNode

ProcessingBlockList          (tmcproto-           method), 75
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_ma ster_lea fnode) $dpMasterLeafNode
                                         read_subarray1HealthState ()          (tmcproto-
                                         type.centralnode.src.centralnode.central_node.CentralNode
                                         method), 3

R
read_activeProcessingBlocks ()          (tmcproto-           method), 53
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.type_cen tral_node_sd pMasterLeafNode central_node_low.CentralNode
                                         read_subarray1HealthState ()          (tmcproto-
                                         type.centralnode_sd pMasterLeafNode central_node_low.CentralNode
                                         method), 8

read_activityMessage ()          (tmcproto-           method), 3
       type.centralnode_sd pMasterLeafNode central_node_low.CentralNode
                                         read_subarray2HealthState ()          (tmcproto-
                                         type.centralnode_sd pMasterLeafNode central_node_low.CentralNode
                                         method), 3

read_activityMessage ()          (tmcproto-           method), 8
       type.centralnode_low.src.centralnode_low.CentralNode
                                         read_subarray3HealthState ()          (tmcproto-
                                         type.centralnode_low.src.centralnode_low.CentralNode
                                         method), 3

read_activityMessage ()          (tmcproto-           method), 8
       type.cspmasterleafnode.src.cspmasterleafnode.csp_ma ster_leafnode) telescop eHealthState ()          (tmcproto-
                                         type.centralnode_sd pMasterLeafNode central_node.CentralNode
                                         method), 3

```

method), 3
 read_telescopeHealthState() (tmcproto- ReleaseResources (class in tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_low.CentralNode
 type.centralnode_low.src.centralnode_low.central_node_low.CentralNode
 method), 8
 read_versionInfo() (tmcproto- ReleaseResources (in module tmcproto-
 type.centralnode.src.centralnode.central_node),
 type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
 method), 76
 receive_addresses_cb() (tmcproto- ReleaseResources () (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 1
 method), 18
 receiveAddresses (tmcproto- ReleaseResources () (tmcproto-
 type.centralnode_low.src.centralnode_low.central_node_low.CentralNode
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
 attribute), 53
 receptorIDList (tmcproto- ReleaseResources () (tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_low.CentralNode
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 83
 attribute), 19
 release_csp_resources() (tmcproto- releaseResources_cmd_ended_cb () (tmcproto-
 type.sdpmasterleafnode.src.sdpmasterleafnode.release_all_resources_method, 88 ReleaseAllResourcesCommand
 method), 21
 remove_receptors_from_group () (tmcproto-
 release_sdp_resources() (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 type.subarraynode.src.subarraynode.release_all_resources_method, 91 ReleaseAllResourcesCommand
 method), 21
 Restart (class in tmcproto-
 ReleaseAllResources (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_command,
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_all_resources_command),
 64
 Restart (class in tmcproto-
 ReleaseAllResources (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node),
 55
 Restart () (tmcproto-
 ReleaseAllResources () (tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
 method), 51
 Restart () (tmcproto-
 releaseallresources_cmd_ended_cb() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_all_resources_command, ReleaseAllResources
 method), 65
 restart_cmd_ended_cb () (tmcproto-
 releaseallresources_cmd_ended_cb() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_command,
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.ReleaseAllResources
 method), 55
 restart_cmd_ended_cb () (tmcproto-
 ReleaseAllResourcesCommand (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 type.subarraynode.src.subarraynode.release_all_resources_command), (class in tmcproto-
 21
 type.subarraynode.src.subarraynode.restart_command),
 24
 ReleaseAllResourcesCommand (class in tmcproto- S_all_resources_command),
 type.subarraynode_low.src.subarraynode_low.release_all_resources_command,
 30
 sbID (tmcproto.type.subarraynode.src.subarraynode.subarray_node.SubarrayNode)
 ReleaseResources (class in tmcproto- attribute), 19
 type.centralnode.src.centralnode.release_resources_command, (class in tmcproto-
 5
 type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray
 ReleaseResources (class in tmcproto- 92
 type.centralnode_low.src.centralnode_low.central_node_low), (class in tmcproto-
 10
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.scan_command
 ReleaseResources (class in tmcproto- 66
 type.centralnode_low.src.centralnode_low.release_resources_command), (class in tmcproto-
 13
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray

```

60                                     SdpSubarrayLeafNode.InitCommand
Scan() (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode in tmcproto-
       method), 35                                         type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_
Scan() (tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode
       method), 89                                         SdpSubarrayLNFQDN (tmcproto-
                                                               method)
Scan() (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode.SubarrayNo
       method), 52                                         attribute), 17
scan_cmd_ended_cb() (tmcproto- sdpSubarrayObsState (tmcproto-
       type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.SdpSubarrayLeafNode.sdp_subarray_
       method), 92                                         attribute), 53
scan_cmd_ended_cb() (tmcproto- set_dish_name_number() (tmcproto-
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.scan_command.DishLeafNode.dish_leaf_node.DishLeafNode
       method), 66                                         attribute), 39
scan_cmd_ended_cb() (tmcproto- set_observer_lat_long_alt() (tmcproto-
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode.dish_leaf_node.DishLeafNode
       method), 60                                         method), 39
ScanCommand (class in tmcproto- SetOperateMode() (tmcproto-
       type.subarraynode.src.subarraynode.scan_command), type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
       22                                         method), 35
ScanCommand (class in tmcproto- SetStandbyFPMode() (tmcproto-
       type.subarraynode_low.src.subarraynode_low.scan_command), type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
       31                                         method), 35
scanID (tmcprototype.subarraynode.src.subarraynode.subarraynode_SetStandbyMode) (tmcproto-
       attribute), 19                                         type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
scanID (tmcprototype.subarraynode_low.src.subarraynode_low.subarraynode_SetStandbyMode) (tmcproto-
       attribute), 28                                         type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
SetStandbyMode() (tmcproto-                                         method), 46
sdpHealthState (tmcproto-                                         type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
       attribute), 76                                         method), 35
SdpMasterFQDN (tmcproto-                                         type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
       attribute), 75                                         method), 36
SdpMasterLeafNode (class in tmcproto-                                         type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
       75                                         class in tmcproto-
                                         type.cspmasterleafnode.src.cspmasterleafnode.standby_command
                                         45
SdpMasterLeafNode.InitCommand (class in tmcproto- Standby (class in tmcproto-
       type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_node.SdpMasterLeafNode
       75                                         type.cspmasterleafnode.src.cspmasterleafnode.standby_command
                                         77
SdpMasterLeafNodeFQDN (tmcproto- Standby (class in tmcproto-
       type.centralnode.src.centralnode.central_node.CentralNode.type.sdpmasterleafnode.src.sdpmasterleafnode.standby_command
       attribute), 1                                         81
SdpSubarrayFQDN (tmcproto- Standby () (tmcproto-
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode.csp_master_leaf_node
       attribute), 52                                         method), 43
SdpSubarrayFQDN (tmcproto- Standby () (tmcproto-
       type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_
       attribute), 17                                         method), 75
sdpSubarrayHealthState (tmcproto- standby_cmd_ended_cb() (tmcproto-
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode.csp_master_leaf_node
       attribute), 53                                         method), 45
SdpSubarrayLeafNode (class in tmcproto- standby_cmd_ended_cb() (tmcproto-
       type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode.cspmasterleafnode.standby_command
       51                                         method), 47

```

standby_cmd_ended_cb() (tmproto- startup_subarray() (tmproto-
 type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_type.cmdendedcallback_low.src.centralnode_low.central_node_low.StartUp
 method), 78 method), 12

standby_cmd_ended_cb() (tmproto- startup_subarray() (tmproto-
 type.sdpmasterleafnode.src.sdpmasterleafnode.standby_commandcallback_low.src.centralnode_low.startup_telescope_commu
 method), 81 method), 15

standby_leaf_node() (tmproto- StartUpTelescope (class in tmproto-
 type.centralnode_low.src.centralnode_low.StandByTelescope_low.src.centralnode_low.central_node_low),
 method), 11 11

standby_leaf_node() (tmproto- StartUpTelescope (class in tmproto-
 type.centralnode_low.src.centralnode_low.standby_telescope_type_and_standby_telescope_low.src.centralnode_low.startup_telescope_commu
 method), 14 15

standby_mccs_mln() (tmproto- StartUpTelescope (in module tmproto-
 type.centralnode_low.src.centralnode_low.StandByTelescope_low.src.centralnode.central_node),
 method), 11 3

standby_mccs_mln() (tmproto- StartUpTelescope () (tmproto-
 type.centralnode_low.src.centralnode_low.standby_telescope_type_and_standby_telescope_low.src.centralnode_low.CentralNode
 method), 14 method), 2

standby_subarray() (tmproto- StartUpTelescope () (tmproto-
 type.centralnode_low.src.centralnode_low.StandByTelescope_low.src.centralnode_low.central_node_low.CentralNode
 method), 11 method), 7

standby_subarray() (tmproto- stop_dish_tracking() (tmproto-
 type.centralnode_low.src.centralnode_low.standby_telescope_type_and_standby_telescope_low.end_command.EndCommand
 method), 15 method), 23

StandByTelescope (class in tmproto- StopCapture() (tmproto-
 type.centralnode_low.src.centralnode_low.central_node_low), type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 10 method), 37

StandByTelescope (class in tmproto- StopTrack() (tmproto-
 type.centralnode_low.src.centralnode_low.standby_telescope_type_and_standby_telescope_low.src.dishleafnode.dish_leaf_node.DishLeafNode
 14 method), 37

StandByTelescope (in module tmproto- StowAntennas (class in tmproto-
 type.centralnode_low.src.centralnode.central_node), type.centralnode_low.stow_antennas_command),
 3 6

StandByTelescope() (tmproto- StowAntennas (in module tmproto-
 type.centralnode_low.src.centralnode.central_node.CentralNode), type.centralnode_low.src.centralnode.central_node.CentralNode
 method), 2 4

StandByTelescope() (tmproto- StowAntennas() (tmproto-
 type.centralnode_low.src.centralnode_low.CentralNode), type.centralnode_low.src.centralnode.central_node.CentralNode
 method), 7 method), 2

StartCapture() (tmproto- subarray1HealthState (tmproto-
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode), type.centralnode_low.src.centralnode.central_node.CentralNode
 method), 36 attribute), 3

startup_leaf_node() (tmproto- subarray1HealthState (tmproto-
 type.centralnode_low.src.centralnode_low.StopUpTelescope_low.src.centralnode_low.central_node_low.CentralNode
 method), 11 attribute), 8

startup_leaf_node() (tmproto- subarray2HealthState (tmproto-
 type.centralnode_low.src.centralnode_low.startup_telescope_type_and_standby_telescope_low.src.centralnode_low.CentralNode
 method), 15 attribute), 3

startup_mccs_mln() (tmproto- subarray3HealthState (tmproto-
 type.centralnode_low.src.centralnode_low.StopUpTelescope_low.src.centralnode.central_node.CentralNode
 method), 11 attribute), 3

startup_mccs_mln() (tmproto- subarray_health_subscribe_event() (tmproto-
 type.centralnode_low.src.centralnode_low.startup_telescope_type_and_standby_telescope_low.health_state_aggreg
 method), 15 method), 13

```

SubarrayNode      (class      in      tmcproto-  tmcprototype.cspmasterleafnode.src.cspmasterleafnode
                  type.subarraynode.src.subarraynode.subarray_node),      (module), 46
                  17                           tmcprototype.cspmasterleafnode.src.cspmasterleafnode

SubarrayNode      (class      in      tmcproto-      (module), 46
                  type.subarraynodelow.src.subarraynodelow.subarray_node_type.dishleafnode.src.dishleafnode.dish_lea
                  27                           (module), 33

SubarrayNode.InitCommand (class in tmcproto-  tmcprototype.mccsmasterleafnode.src.mccsmasterleafn
                  type.subarraynode.src.subarraynode.subarray_node),      (module), 83
                  17                           tmcprototype.mccssubarrayleafnode.src.mccssubarrayl

SubarrayNode.InitCommand (class in tmcproto-      (module), 89
                  type.subarraynodelow.src.subarraynodelow.subarray_node_type.sdpmasterleafnode.src.sdpmasterleafn
                  27                           (module), 79

subscribe_event ()          (tmcproto-  tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode
                  type.centralnodelow.src.centralnodelow.health_state_aggregatorHealthStateAggregator
                  method), 13                           (module), 80
                                              tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode
                                              (module), 80

T

telescopeHealthState          (tmcproto-  tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode
                  type.centralnode.src.centralnode.central_node.CentralNode
                  attribute), 3                           (module), 75
                                              tmcprototype.sdpmasterleafnode.src.sdpmasterleafnode
                                              (module), 81

telescopeHealthState          (tmcproto-  tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafn
                  type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
                  attribute), 8                           (module), 68
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf
                                              (module), 68

TMAlarmHandler          (tmcproto-  tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf
                  type.centralnode.src.centralnode.central_node.CentralNode
                  attribute), 2                           (module), 63
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf
                                              (module), 65

TMAlarmHandler          (tmcproto-  tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf
                  type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
                  attribute), 8                           (module), 68
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf
                                              (module), 68

tmcprototype.centralnode.src.centralnode.assign_resources_command
                  (module), 4                           (module), 67
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnode.src.centralnode.central_node
                  (module), 1                           (module), 70
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnode.src.centralnode.release_resources_command
                  (module), 5                           (module), 63
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnode.src.centralnode.stow_antennas_command
                  (module), 6                           (module), 62
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnodelow.src.centralnodelow.assign_resources_command
                  (module), 12                          (module), 64
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnodelow.src.centralnodelow.node_low
                  (module), 7                           (module), 69
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnodelow.src.centralnodelow.state_aggregator
                  (module), 13                          (module), 66
                                              tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleaf

tmcprototype.centralnodelow.src.centralnodelow.release_resources_command
                  (module), 13                          (module), 51
                                              tmcprototype.subarraynode.src.subarraynode.abort_c
                                              (module), 13

tmcprototype.centralnodelow.src.centralnodelow.telescope_command
                  (module), 14                          (module), 23
                                              tmcprototype.subarraynode.src.subarraynode.assign_r
                                              (module), 14

tmcprototype.centralnodelow.src.centralnodelow.telescope_command
                  (module), 15                          (module), 20
                                              tmcprototype.subarraynode.src.subarraynode.configu
                                              (module), 15

tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node
                  (module), 43                          (module), 22
                                              tmcprototype.subarraynode.src.subarraynode.end_com
                                              (module), 43

tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_command
                  (module), 45                          (module), 23
                                              tmcprototype.subarraynode.src.subarraynode.end_sc
                                              (module), 45
                                              (module), 23

```

tmcprototype.subarraynode.src.subarraynode.observe_command
 (module), 24
 update_configuration_json() (tmcproto-
 type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
 method), 91
 tmcprototype.subarraynode.src.subarraynode.on_command
 (module), 19
 V
 tmcprototype.subarraynode.src.subarraynode.release_all_resources_command
 (module), 21
 tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 method), 91
 tmcprototype.subarraynode.src.subarraynode.restore_command
 (module), 24
 validate_obs_state() (tmcproto-
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 method), 19
 tmcprototype.subarraynode.src.subarraynode.set_subarray_node
 (module), 17
 tmcproto-
 type.sdpmasterleafnode.src.sdpmasterleafnode.sdp_master_leaf_n
 attribute), 76
 tmcprototype.subarraynode.src.subarraynode.track_command
 (module), 24
 W
 tmcprototype.subarraynodelow.src.subarraynodelow.assign_resources_command
 (module), 29
 write_activityMessage() (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
 tmcprototype.subarraynodelow.src.subarraynodelow_low.end_command
 (module), 29
 write_activityMessage() (tmcproto-
 type.centralnodelow.src.centralnodelow.central_node_low.Central
 Method), 8
 tmcprototype.subarraynodelow.src.subarraynodelow_low_end_scan_command
 (module), 30
 write_activityMessage() (tmcproto-
 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_n
 method), 44
 tmcprototype.subarraynodelow.src.subarraynodelow_low_on_command
 (module), 30
 write_activityMessage() (tmcproto-
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 Method), 39
 tmcprototype.subarraynodelow.src.subarraynodelow_low_release_all_resources_command
 (module), 30
 write_activityMessage() (tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
 method), 84
 tmcprototype.subarraynodelow.src.subarraynodelow_low_subarray_node_low
 (module), 27
 type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_suba
 method), 90
 TMLowSubarrayNodes
 (tmcproto-
 type.centralnodelow.src.centralnode_low.central_node_low.CentralNode
 attribute), 8
 TMMidSubarrayNodes
 (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 attribute), 2
 Track()
 (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 method), 37
 write_activityMessage() (tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 method), 35
 Track()
 (tmcprototype.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 method), 17
 write_activityMessage() (tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l
 method), 19
 track_thread()
 (tmcproto-
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 method), 39
 write_activityMessage() (tmcproto-
 type.subarraynodelow.src.subarraynodelow_low.subarray_node_low.S
 method), 28
 TrackCommand
 (class in tmcproto-
 type.subarraynode.src.subarraynode.track_command),
 write_receiveAddresses() (tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 method), 53
 24
 U
 unsubscribe_event()
 (tmcproto-
 type.centralnodelow.src.centralnode_low.health_state_aggregator.HealthStateAggregator