
TMC Prototype Documentation

Release 1.0

NCRA India

Jan 05, 2021

Contents:

1 Central Node	1
2 Central Node	7
3 Subarray Node	13
4 Subarray Node	23
5 Dish Leaf Node	29
6 Dish Master	37
7 CSP Master Leaf Node	39
8 SDP Subarray Leaf Node	45
9 Subarray Node	47
10 CSP Subarray Leaf Node	69
11 SDP Master Leaf Node	71
12 MCCS Master Leaf Node	73
13 MCCS Subarray Leaf Node	79
14 Indices and tables	85
Python Module Index	87
Index	89

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 `tmctype.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 SDPMasterLeafNode.

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 CspMasterLeafNode and SdpMasterLeafNode and Off() command on SubarrayNode and sets CentralNode into OFF state.

StartUpTelescope ()

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

StowAntennas (argin)

This command stows the specified receptors.

TMAAlarmHandler

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":"02:42:4","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", "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 not released from the subarray.

Example: argout = {
 "ReleaseAll" : True, "receptorIDList" : []
}

rtype None

raises ValueError if input argument json string contains invalid value Key-Error 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 tmcprototype.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

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.centralnodelow.src.centralnodelow.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.

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

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

Returns None

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

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

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.

class ReleaseResourcesCommand (*args, **kwargs)

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, 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 Key-Error if input argument json string contains invalid key DevFailed if the command execution or command invocation on SubarrayNode is not successful

StandByTelescope ()

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

class StandByTelescopeCommand (*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)

Raises DevFailed if error occurs while invoking command on any of the devices like SubarrayNode or MccsMasterLeafNode.

StartupTelescope ()

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

class StartupTelescopeCommand (*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)

Raises DevFailed if error occurs while invoking command on any of the devices like SubarrayNode or MccsMasterLeafNode.

TMAAlarmHandler

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.

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

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.

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”:“visibility”}],{“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.ReleaseAllResourcesCommand
```

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":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'], [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]]}} "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'], [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`)

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.subarraynodelow.src.subarraynodelow.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.subarraynodelow.src.subarraynodelow.assign_resources_command.AssignResourcesCommand
```

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.subarraynodelow.src.subarraynodelow.configure_command.ConfigureCommand (args=None,
```

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”: [ {“station_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.subarraynodelow.src.subarraynodelow.end_command.EndCommand (*args,
                                                                    **kwargs)
```

A class for SubarrayNodeLow's End() command.

do ()

This command on Subarray Node Low invokes End command on MCCA 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 mccs 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
```

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 MCCS 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

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 DishMaster.

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 DishMaster

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 DishMaster.

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 DishMaster.

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 DishMaster.

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

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.CspMasterL
```

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 events.

```
Standby (arg)
```

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

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.SdpS

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=None, **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.**Release**

A class for SdpSubarayLeafNode'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** (*C

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.ObsR`

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_command.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

AssignResouces class for SDPSubarrayLeafNode.

class tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.assign_resources_command.**As**

A class for SdpSubarayLeafNode'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”:“dprep”,“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.

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` (*arg
**kw

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 (*C
**

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

MCCS Master Leaf Node

class tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.**MccsMasterLeafNode**

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 AssignResourcesCommand (**args, **kwargs*)

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 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

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]
```

}

Returns None

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

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 InitCommand (*args, **kwargs)

A class for the TMC MCCA 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 events.

MccsMasterFQDN

Used by autodoc_mock_imports.

class OffCommand (*args, **kwargs)

A class for MccsMasterLeafNode's Off() command.

do ()

Invokes Off command on the MCCA 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)

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

class OnCommand (*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

ReleaseResources (argin)

Invokes ReleaseResources command on MccsMasterLeafNode

class ReleaseResourcesCommand (*args, **kwargs)

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

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.

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`

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

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

do (*argin*)

This command configures a scan. It accepts configuration information in JSON string format and invokes Configure command on MccsSubarray.

:param argin:DevString. The string in JSON format. The JSON contains following values:

Example: {"stations":[{"station_id":1},{ "station_id":2}], "station_beam_pointings":[{"station_beam_id":1,"target":

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

End ()

Invokes End command on MccsSubarrayLeafNode.

class EndCommand (**args, **kwargs*)

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

EndScan ()

Invokes EndScan command on MccsSubarray.

class EndScanCommand (*args, **kwargs)

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 InitCommand (*args, **kwargs)

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 mccsubarrayleafnode

class ScanCommand (**args, **kwargs*)

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

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.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node.**main** (arg
**k

CHAPTER 14

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode 60
tmcprototype.centralnode.src.centralnode.assign_resources_command,
4 tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode 65
tmcprototype.centralnode.src.centralnode.central_node,
1 tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode 62
tmcprototype.centralnode.src.centralnode.release_resources_command,
5 tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode 47
tmcprototype.centralnode.src.centralnode.stow_antennas_command,
6 tmcprototype.subarraynode.src.subarraynode.abort_command 19
tmcprototype.centralnodelow.src.centralnodelow.central_node_low,
7 tmcprototype.subarraynode.src.subarraynode.assign_resources_command 16
tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node,
39 tmcprototype.subarraynode.src.subarraynode.configure_command 18
tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_command,
41 tmcprototype.subarraynode.src.subarraynode.end_command 19
tmcprototype.cspmasterleafnode.src.cspmasterleafnode.on_command,
42 tmcprototype.subarraynode.src.subarraynode.end_scan_command 19
tmcprototype.cspmasterleafnode.src.cspmasterleafnode.standby_command,
42 tmcprototype.subarraynode.src.subarraynode.obsreset_command 20
tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node,
29 tmcprototype.subarraynode.src.subarraynode.off_command 15
tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node,
73 tmcprototype.subarraynode.src.subarraynode.on_command 15
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.abort_command,
64 tmcprototype.subarraynode.src.subarraynode.release_resources_command 17
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.assign_resources_command,
59 tmcprototype.subarraynode.src.subarraynode.restart_command 20
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.configure_command,
61 tmcprototype.subarraynode.src.subarraynode.scan_command 18
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.end_command,
64 tmcprototype.subarraynode.src.subarraynode.subarraynode 13
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.endscan_command,
63 tmcprototype.subarraynode.src.subarraynode.track_command 20
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset_command,
66 tmcprototype.subarraynodelow.src.subarraynodelow.assign_resources_command 25
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command,
59 tmcprototype.subarraynodelow.src.subarraynodelow.configure_command 25
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_command,
58 tmcprototype.subarraynodelow.src.subarraynodelow.end_command 25

tmcprototype.subarraynodelow.src.subarraynodelow.end_scan_command,
26

tmcprototype.subarraynodelow.src.subarraynodelow.off_command,
26

tmcprototype.subarraynodelow.src.subarraynodelow.on_command,
26

tmcprototype.subarraynodelow.src.subarraynodelow.release_all_resources_command,
26

tmcprototype.subarraynodelow.src.subarraynodelow.scan_command,
27

tmcprototype.subarraynodelow.src.subarraynodelow.subarray_node_low,
23

A

Abort (class in tmcproto-activityMessage (tmcproto-attribute), 82
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.abort_command), 64

Abort (class in tmcproto-attribute), 48
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_node), 53

Abort () (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode (tmcproto-attribute), 43
method), 29

Abort () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_node_low.SdpSubarrayLeafNode (tmcproto-attribute), 23
method), 47

abort_cmd_ended_cb () (tmcproto-add_receptors_in_group () (tmcproto-attribute), 16
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.abort_command.AbortCommand), 64

abort_cmd_ended_cb () (tmcproto-allocate_ended () (tmcproto-attribute), 73
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_node_low.SdpSubarrayLeafNode), 53

AbortCommand (class in tmcproto-always_executed_hook () (tmcproto-attribute), 2
type.subarraynode.src.subarraynode.abort_command), 19

activeProcessingBlocks (tmcproto-always_executed_hook () (tmcproto-attribute), 48
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_node_low.SdpSubarrayLeafNode), 10

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 2
type.centralnode.src.centralnode.central_node.CentralNode), 39

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 10
type.centralnode_low.src.centralnode_low.central_node_low.CentralNode), 76

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 39
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode), 82

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 34
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode), 48

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 76
type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMasterLeafNode), 14

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 14
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode), 14

activityMessage (tmcproto-always_executed_hook () (tmcproto-attribute), 14
type.subarraynode_low.src.subarraynode_low.subarray_node_low.SdpSubarrayLeafNode), 14

method), 23

assign_csp_resources() (tmcproto- build_up_sdp_cmd_data() (tmcproto- type.subarraynode.src.subarraynode.configure_command.ElementDerivedData static method), 18

assign_resources_command.AssignResourcesCommand method), 16

assign_sdp_resources() (tmcproto- **C** calculate_observation_state() (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode method), 16

AssignResources (class in tmcproto- method), 14

AssignResources (class in tmcproto- calculate_observation_state() (tmcproto- type.subarraynode.src.subarraynode.subarray_node_low.SubarrayNode method), 23

AssignResources (class in tmcproto- assign_resources_command() (tmcproto- type.sdp_subarray_leaf_node.src.sdp_subarray_leaf_node.assign_resources_command() method), 59

AssignResources (class in tmcproto- method), 18

AssignResources (class in tmcproto- sdp_subarray_leaf_command() (tmcproto- type.sdp_subarray_leaf_node.src.sdp_subarray_leaf_node.sdp_subarray_leaf_command() method), 50

AssignResources (in module tmcproto- method), 27

AssignResources (type.centralnode.src.centralnode.central_node), CentralAlarmHandler (tmcproto- type.centralnode.src.centralnode.central_node.CentralNode attribute), 1

AssignResources (type.centralnode.src.centralnode.central_node.CentralNode method), 1

AssignResources (type.centralnode_low.src.centralnode_low.central_node_low.CentralNode attribute), 8

AssignResources (type.centralnode_low.src.centralnode_low.central_node_low.CentralNode (class in tmcproto- type.centralnode.src.centralnode.central_node), method), 7

AssignResources (type.mccs_master_leaf_node.src.mccs_master_leaf_node.MccsMasterLeafNode tmcproto- type.centralnode_low.src.centralnode_low.central_node_low), method), 73

AssignResources (type.sdp_subarray_leaf_node.src.sdp_subarray_leaf_node.SdpSubarrayLeafNode tmcproto- type.centralnode_low.src.centralnode_low.central_node_low), method), 47

AssignResources_ended() (tmcproto- type.centralnode_low.src.centralnode_low.central_node_low), method), 59

AssignResources_ended() (tmcproto- type.sdp_subarray_leaf_node.src.sdp_subarray_leaf_node.assign_resources_command.AssignResources method), 50

AssignResourcesCommand (class in tmcproto- type.centralnode_low.src.centralnode_low.central_node_low), method), 16

AssignResourcesCommand (class in tmcproto- type.subarraynode_low.src.subarraynode_low.assign_resources_command), method), 25

attribute_event_handler() (tmcproto- CentralNode.StandByTelescopeCommand (class in tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode method), 34

B CentralNode.StartupTelescopeCommand (class in tmcproto- type.subarraynode_low.src.subarraynode_low.configure_command.ElementDerivedData static method), 18

build_up_csp_cmd_data() (tmcproto- type.subarraynode_low.src.centralnode_low.central_node_low), method), 18

build_up_dsh_cmd_data() (tmcproto- check_allowed() (tmcproto- type.subarraynode.src.subarraynode.configure_command.ElementDerivedData static method), 18

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnode.src.centralnode.release_resources_command.ReleaseResourcesCommand.src.dishleafnode.dish_leaf_node.DishLeafNode.
method), 5 method), 33

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnode.src.centralnode.stow_antennas_command.StowAntennasCommand.src.masterleafnode.src.mccsmasterleafnode.mccs_master_l
method), 6 method), 74

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLowAssignResourcesCommand.src.masterleafnode.mccs_master_l
method), 7 method), 75

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLowReleaseResourcesCommand.src.subarrayleafnode.mccs_suba
method), 8 method), 79

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLowScanByTiltScopeCommand.src.subarrayleafnode.mccs_suba
method), 9 method), 80

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLowSartByTiltScopeCommand.src.subarrayleafnode.mccs_suba
method), 9 method), 81

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.StandbyCommand.src.mccssubarrayleafnode.mccs_suba
method), 41 method), 82

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.cspmasterleafnode.src.cspmasterleafnode.standby_command.Command.src.sdpsubarrayleafnode.abort_comma
method), 42 method), 65

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.AbortCommand.src.sdpsubarrayleafnode.assign_resour
method), 29 method), 59

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.ConfigureCommand.src.sdpsubarrayleafnode.configure_com
method), 29 method), 61

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.EndScanCommand.src.sdpsubarrayleafnode.end_comman
method), 30 method), 64

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.EndScanCommand.src.sdpsubarrayleafnode.endscan_com
method), 30 method), 63

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.ObsResetCommand.src.sdpsubarrayleafnode.obsreset_com
method), 31 method), 66

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.ReleaseResourcesCommand.src.sdpsubarrayleafnode.release_resou
method), 31 method), 60

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.ResetCommand.src.sdpsubarrayleafnode.restart_comm
method), 32 method), 65

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.ScanCommand.src.sdpsubarrayleafnode.scan_comman
method), 32 method), 62

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.SwpCaptureCommand.src.sdpsubarrayleafnode.sdp_subarray
method), 33 method), 54

check_allowed() (tmcproto- check_allowed() (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode.SwpTrackCommand.src.sdpsubarrayleafnode.sdp_subarray
method), 33 method), 50

check_allowed() (tmcproto- ConfigureCommand (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.ConfigureCommand), method), 53 18

check_allowed() (tmcproto- ConfigureCommand (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.Elbow.src.subarraynodelow.configure_command), method), 56 25

check_allowed() (tmcproto- convert_radec_to_azel() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.ErdStishleafnode.dish_leaf_node.DishLeafNode method), 57 method), 34

check_allowed() (tmcproto- cspHealthState (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.CspHealthState method), 55 attribute), 39

check_allowed() (tmcproto- CspMasterFQDN (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.FQDN method), 51 attribute), 39

check_allowed() (tmcproto- CspMasterLeafNode (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.FQDN method), 54 39

check_allowed() (tmcproto- CspMasterLeafNode.InitCommand (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.Scain method), 56 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode method), 39

check_allowed() (tmcproto- CspMasterLeafNode.FQDN (tmcproto- type.subarraynode.src.subarraynode.track_command_mask_continue_node.FQDN method), 20 type.centralnode.src.centralnode.central_node.CentralNode attribute), 1

command_class_object() (tmcproto- CspMasterLeafNode.FQDN (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode method), 14 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode attribute), 13

command_class_object() (tmcproto- CspMasterLeafNode.FQDN (tmcproto- type.subarraynodelow.src.subarraynodelow.subarray_nodelow.SubarrayNode method), 23 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode attribute), 13

Configure (class in tmcproto- ConfigureCommand (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.configure_command), method), 61

Configure (class in tmcproto- delete_device() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.src.centralnode.central_node.CentralNode method), 53 method), 2

Configure() (tmcproto- delete_device() (tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode method), 29 type.centralnodelow.src.centralnodelow.central_node_low.CentralNode method), 10

Configure() (tmcproto- delete_device() (tmcproto- type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode method), 79 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode method), 40

Configure() (tmcproto- delete_device() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.SdpSubarrayLeafNode method), 47 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMasterLeafNode method), 76

configure_cmd_ended_cb() (tmcproto- delete_device() (tmcproto- type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode method), 79 type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_configure_command.MccsConfigureCommand method), 82

configure_cmd_ended_cb() (tmcproto- delete_device() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.configure_command.Configure method), 61 type.sdpsubarrayleafnode.sdp_subarrayleafnode.Configure method), 48

configure_cmd_ended_cb() (tmcproto- delete_device() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.Configure method), 53 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode method), 14

D

delete_device() (tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node),
type.subarraynode.low.src.subarraynode.low.subarray_node_low.SubarrayNode
method), 23 DishLeafNode.StartCaptureCommand

DeviceData (in module tmcproto- (class in tmcproto-
type.centralnode.src.centralnode.central_node), type.dishleafnode.src.dishleafnode.dish_leaf_node),
3 32

dishHealthState (tmcproto- DishLeafNode.StopCaptureCommand
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode attribute), 34 (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 33

DishLeafNode (class in tmcproto- DishLeafNode.StopTrackCommand
type.dishleafnode.src.dishleafnode.dish_leaf_node), 29 (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 33

DishLeafNode.AbortCommand (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 29 DishLeafNode.TrackCommand (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 33

DishLeafNode.ConfigureCommand (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 29 DishLeafNodePrefix (tmcproto-
type.centralnode.src.centralnode.central_node.CentralNode
attribute), 1

DishLeafNode.EndScanCommand (class in tmcproto- DishLeafNodePrefix (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 30 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
attribute), 13

DishLeafNode.InitCommand (class in tmcproto- DishMasterFQDN (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 30 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
attribute), 30

DishLeafNode.ObsResetCommand (class in tmcproto- dishPointingState (tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node), 30 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
attribute), 34

DishLeafNode.RestartCommand do () (tmcprototype.centralnode.src.centralnode.assign_resources_command
method), 4
(class in tmcproto- do () (tmcprototype.centralnode.src.centralnode.central_node.CentralNode
type.dishleafnode.src.dishleafnode.dish_leaf_node), method), 1
31 do () (tmcprototype.centralnode.src.centralnode.release_resources_command
method), 5

DishLeafNode.ScanCommand (class in tmcproto- do () (tmcprototype.centralnode.src.centralnode.stow_antennas_command
type.dishleafnode.src.dishleafnode.dish_leaf_node), method), 6
31 do () (tmcprototype.centralnode.src.centralnode_low.src.centralnode_low.central_node_low.CentralNodeLow
method), 7

DishLeafNode.SetOperateModeCommand do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.CentralNodeLow
method), 8
(class in tmcproto- do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.CentralNodeLow
type.dishleafnode.src.dishleafnode.dish_leaf_node), method), 8
31 do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.CentralNodeLow
method), 9

DishLeafNode.SetStandbyFPModeCommand do () (tmcprototype.centralnode_low.src.centralnode_low.central_node_low.CentralNodeLow
method), 9
(class in tmcproto- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
type.dishleafnode.src.dishleafnode.dish_leaf_node), method), 39
31 do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
method), 41

DishLeafNode.SetStowModeCommand do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
method), 40
(class in tmcproto- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
type.dishleafnode.src.dishleafnode.dish_leaf_node), method), 40
32 do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
method), 40

DishLeafNode.SlewCommand (class in tmcproto- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
method), 40

- method), 41
- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_comprand), 42
- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.off_comprand), 42
- do () (tmcprototype.cspmasterleafnode.src.cspmasterleafnode.abort_c), 42
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.AbortComleafnode.src.sdpsubarrayleafnode.assign_ method), 29
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.CsconfigureCleafnode.src.sdpsubarrayleafnode.configu method), 29
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.EndScanCleafnode.src.sdpsubarrayleafnode.end_con method), 30
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.EndScanLeafnode.src.sdpsubarrayleafnode.endscar method), 30
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.ObsResetCleafnode.src.sdpsubarrayleafnode.obsrese method), 30
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.RestartComleafnode.src.sdpsubarrayleafnode.off_com method), 31
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSubComleafnode.src.sdpsubarrayleafnode.on_com method), 31
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsOperateMafhaComleafnode.src.sdpsubarrayleafnode.release method), 31
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsStandbyFBModeComleafnode.src.sdpsubarrayleafnode.restart method), 31
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsStandbyLBModeComleafnode.src.sdpsubarrayleafnode.scan_co method), 32
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsShowMafhaModeComleafnode.src.sdpsub method), 32
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSwComleafnode.src.sdpsubarrayleafnode.sdp_sub method), 32
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSubCapbyteCleafnode.src.sdpsubarrayleafnode.sdp_sub method), 32
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSubCapbyteCleafnode.src.sdpsubarrayleafnode.sdp_sub method), 33
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSubBarchCleafnode.src.sdpsubarrayleafnode.sdp_sub method), 33
- do () (tmcprototype.dishleafnode.src.dishleafnode.dish_leafnode(DisplayType.SdsSubComleafnode.src.sdpsubarrayleafnode.sdp_sub method), 33
- do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleAssignLeafnode.sdsCont method), 74
- do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleCleafnode.sdp_sub method), 74
- do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleOffCleafnode.sdp_sub method), 74
- do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleOnCleafnode.sdp_sub method), 75
- do () (tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleRedusaReadausdsSub method), 76
- do () (tmcprototype.mccsubarrayleafnode.src.mccsubarrayleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleCleafnode.sdp_sub method), 80
- do () (tmcprototype.mccsubarrayleafnode.src.mccsubarrayleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleCleafnode.sdp_sub method), 80
- do () (tmcprototype.mccsubarrayleafnode.src.mccsubarrayleafnode.(tmcprototype.sds_subnode.MafhaMode.SteaffisuleCleafnode.sdp_sub method), 19

```

        method), 16
do () (tmcprototype.subarraynode.src.subarraynode.configure_command_and_configure_command
        method), 18
do () (tmcprototype.subarraynode.src.subarraynode.end_command.EndCommand (class in tmcproto-
        method), 19
do () (tmcprototype.subarraynode.src.subarraynode.end_scan_command.EndScanCommand (tmcproto-
        method), 19
do () (tmcprototype.subarraynode.src.subarraynode.obsreset_command.ObserveResetCommand (tmcproto-
        method), 20
do () (tmcprototype.subarraynode.src.subarraynode.off_command.OffCommand (tmcprototype.subarraynode.src.subarraynode.sdp_subarray_
        method), 15
do () (tmcprototype.subarraynode.src.subarraynode.on_command.OnCommand (class in tmcproto-
        method), 15
do () (tmcprototype.subarraynode.src.subarraynode.release_all_resources_command.ReleaseAllResourcesCommand
        method), 17
do () (tmcprototype.subarraynode.src.subarraynode.restart_command.RestartCommand (class in tmcproto-
        method), 20
do () (tmcprototype.subarraynode.src.subarraynode.scan_command.ScanCommand (class in tmcproto-
        method), 19
do () (tmcprototype.subarraynode.src.subarraynode.subarray_node.SubarrayNode.InitCommand
        method), 13
do () (tmcprototype.subarraynode.src.subarraynode.track_command.TrackCommand (class in tmcproto-
        method), 20
do () (tmcprototype.subarraynodelow.src.subarraynodelow.assign_resources_command.AssignResourcesCommand (tmcprototype.subarraynodelow.src.subarraynodelow.dish_leaf_node.DishLeafNode
        method), 25
do () (tmcprototype.subarraynodelow.src.subarraynodelow.configure_command_and_configure_command
        method), 25
do () (tmcprototype.subarraynodelow.src.subarraynodelow.end_command.EndCommand (class in tmcproto-
        method), 25
do () (tmcprototype.subarraynodelow.src.subarraynodelow.end_scan_command.EndScanCommand (tmcproto-
        method), 26
do () (tmcprototype.subarraynodelow.src.subarraynodelow.off_command.OffCommand (tmcprototype.subarraynodelow.src.subarraynodelow.endscan_cmd_ended_cb () (tmcproto-
        method), 26
do () (tmcprototype.subarraynodelow.src.subarraynodelow.on_command.OnCommand (class in tmcproto-
        method), 26
do () (tmcprototype.subarraynodelow.src.subarraynodelow.release_all_resources_command.ReleaseAllResourcesCommand (tmcprototype.subarraynodelow.src.subarraynodelow.endscan_com
        method), 26
do () (tmcprototype.subarraynodelow.src.subarraynodelow.scan_command.ScanCommand (class in tmcproto-
        method), 27
do () (tmcprototype.subarraynodelow.src.subarraynodelow.subarray_node.SubarrayNode.InitCommand (class in tmcprototype.subarraynodelow.src.subarraynodelow.sdp_subarray_
        method), 23
E
ElementDeviceData (class in tmcproto-
        type.subarraynode.src.subarraynode.configure_command_and_configure_command),
        18
End (class in tmcproto-
        type.sdp_subarrayleafnode.src.sdp_subarrayleafnode_end_command),
        64
G
End (class in tmcproto-
        type.sdp_subarrayleafnode.src.sdp_subarrayleafnode.sdp_subarray_node.SubarrayNode
        method), 14
End () (tmcprototype.mccsubarrayleafnode.src.mccsubarrayleafnode.mccsubarray_leaf_node.MccSubarrayLeafNode (class in tmcprototype.subarraynodelow.src.subarraynodelow.subarray_node_low.S
        method), 80

```

	<i>method</i>), 24		<code>is_AssignResources_allowed()</code>	(<i>tmcproto-</i> <i>type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l</i> <i>method</i>), 76
H				
<code>health_state_cb()</code>	(<i>tmcproto-</i> <i>type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLow</i> <i>method</i>), 10		<code>is_AssignResources_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 48
<code>health_state_cb()</code>	(<i>tmcproto-</i> <i>type.subarraynode.src.subarraynode.subarray_node.SubarrayNode</i> <i>method</i>), 14		<code>is_Configure_allowed()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34
<code>health_state_cb()</code>	(<i>tmcproto-</i> <i>type.subarraynodelow.src.subarraynodelow.subarray_node_low.SubarrayNodeLow</i> <i>method</i>), 24		<code>is_Configure_allowed()</code>	(<i>tmcproto-</i> <i>type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_suba</i> <i>method</i>), 83
I			<code>is_Configure_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 48
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.centralnode.src.centralnode.central_node.CentralNode</i> <i>method</i>), 2		<code>is_End_allowed()</code>	(<i>tmcproto-</i> <i>type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_suba</i> <i>method</i>), 83
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLow</i> <i>method</i>), 10		<code>is_End_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 48
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode</i> <i>method</i>), 40		<code>is_EndScan_allowed()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34		<code>is_EndScan_allowed()</code>	(<i>tmcproto-</i> <i>type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_suba</i> <i>method</i>), 83
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMasterLeafNode</i> <i>method</i>), 76		<code>is_EndScan_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 48
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode</i> <i>method</i>), 82		<code>is_ObsReset_allowed()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode</i> <i>method</i>), 48		<code>is_ObsReset_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 48
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.subarraynode.src.subarraynode.subarray_node.SubarrayNode</i> <i>method</i>), 14		<code>is_ReleaseAllResources_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarra</i> <i>method</i>), 48
<code>init_command_objects()</code>	(<i>tmcproto-</i> <i>type.subarraynodelow.src.subarraynodelow.subarray_node_low.SubarrayNodeLow</i> <i>method</i>), 24		<code>is_ReleaseResources_allowed()</code>	(<i>tmcproto-</i> <i>type.centralnode.src.centralnode.central_node.CentralNode</i> <i>method</i>), 2
<code>is_Abort_allowed()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34		<code>is_ReleaseResources_allowed()</code>	(<i>tmcproto-</i> <i>type.centralnodelow.src.centralnodelow.central_node_low.Centra</i> <i>method</i>), 10
<code>is_Abort_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode</i> <i>method</i>), 48		<code>is_ReleaseResources_allowed()</code>	(<i>tmcproto-</i> <i>type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_l</i> <i>method</i>), 76
<code>is_AssignResources_allowed()</code>	(<i>tmcproto-</i> <i>type.centralnode.src.centralnode.central_node.CentralNode</i> <i>method</i>), 2		<code>is_Restart_allowed()</code>	(<i>tmcproto-</i> <i>type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode</i> <i>method</i>), 34
<code>is_AssignResources_allowed()</code>	(<i>tmcproto-</i> <i>type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLow</i> <i>method</i>), 10		<code>is_Restart_allowed()</code>	(<i>tmcproto-</i> <i>type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray</i> <i>method</i>), 49

is_Scan_allowed() (tmcproto- type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 main() (in module tmcproto-

is_Scan_allowed() (tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node),
type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode
method), 83 main() (in module tmcproto-

is_Scan_allowed() (tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
method), 49 main() (in module tmcproto-

is_Slew_allowed() (tmcproto- type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 main() (in module tmcproto-

is_Standby_allowed() (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
method), 40 main() (in module tmcproto-

is_StandByTelescope_allowed() (tmcproto- type.subarraynode.src.subarraynode.subarray_node),
type.centralnode.src.centralnode.central_node.CentralNode
method), 2 main() (in module tmcproto-

is_StandByTelescope_allowed() (tmcproto- type.subarraynodelow.src.subarraynodelow.subarray_node_low.CentralNode
type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
method), 10 mccsHealthState (tmcproto-

is_StartCapture_allowed() (tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 MccsMasterFQDN (tmcproto-

is_StartUpTelescope_allowed() (tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.centralnode.src.centralnode.central_node.CentralNode
method), 2 MccsMasterLeafNode (class in tmcproto-

is_StartUpTelescope_allowed() (tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
method), 11 MccsMasterLeafNode.AssignResourcesCommand

is_StopCapture_allowed() (tmcproto- (class in tmcproto-
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 MccsMasterLeafNode (class in tmcproto-

is_StopTrack_allowed() (tmcproto- MccsMasterLeafNode.InitCommand
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 MccsMasterLeafNode (class in tmcproto-

is_StowAntennas_allowed() (tmcproto- MccsMasterLeafNode.OffCommand
type.centralnode.src.centralnode.central_node.CentralNode
method), 3 MccsMasterLeafNode (class in tmcproto-

is_Track_allowed() (tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35 MccsMasterLeafNode.OnCommand

is_Track_allowed() (tmcproto- (class in tmcproto-
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
method), 14 MccsMasterLeafNode (class in tmcproto-

M MccsMasterLeafNode.ReleaseResourcesCommand
(class in tmcproto-

main() (in module tmcproto- type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.
type.centralnode.src.centralnode.central_node), 3 MccsMasterLeafNodeFQDN (tmcproto-

main() (in module tmcproto- type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
type.centralnodelow.src.centralnodelow.central_node_low), attribute), 8
11 MccsSubarrayFQDN (tmcproto-

main() (in module tmcproto- type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.

attribute), 81
MccsSubarrayFQDN (tmcproto- ObsReset () (tmcproto-
type.subarraynode.low.src.subarraynode.low.subarray_node_type.SubarrayNode
attribute), 23 method), 30
mccsSubarrayHealthState (tmcproto- ObsReset () (tmcproto-
type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode
attribute), 83 method), 47
MccsSubarrayLeafNode (class in tmcproto- obsreset_cmd_ended_cb () (tmcproto-
type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode
79 method), 66
MccsSubarrayLeafNode.ConfigureCommand (class in tmcproto- obsreset_cmd_ended_cb () (tmcproto-
type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node),
79 ObsResetCommand (class in tmcproto-
MccsSubarrayLeafNode.EndCommand (class in tmcproto- type.subarraynode.src.subarraynode.obsreset_command),
20
type.mccsubarrayleafnode.src.mccsubarrayleafnode.MccsSubarrayLeafNode in module tmcproto-
80 type.centralnode.src.centralnode.central_node),
MccsSubarrayLeafNode.EndScanCommand 3
(class in tmcproto- Off (class in tmcproto-
type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node),
81 type.cspmaterleafnode.src.cspmaterleafnode.csp_master_leaf_node),
41
MccsSubarrayLeafNode.InitCommand Off (class in tmcproto-
(class in tmcproto- type.cspmaterleafnode.src.cspmaterleafnode.off_command),
type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leaf_node),
81 4
Off (class in tmcproto-
MccsSubarrayLeafNode.ScanCommand type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command
(class in tmcproto- 59
type.mccsubarrayleafnode.src.mccsubarrayleafnode.MccsSubarrayLeafNode in tmcproto-
82 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdpsubarray_LeafNode),
MccsSubarrayLNFQDN (tmcproto- 52
type.subarraynode.low.src.subarraynode.low.subarray_node.LowSubarrayNode (tmcproto-
attribute), 23 type.mccsmaterleafnode.src.mccsmaterleafnode.mccs_master_leaf_node),
mccsSubarrayObsState (tmcproto- method), 74
type.mccsubarrayleafnode.src.mccsubarrayleafnode.MccsSubarrayLeafNode
attribute), 83 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.off_command.
method), 59
N off_cmd_ended_cb () (tmcproto-
NumDishes (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdpsubarray_LeafNode
type.centralnode.src.centralnode.central_node.CentralNode method), 52
attribute), 1 OffCommand (class in tmcproto-
15
type.subarraynode.src.subarraynode.off_command),
O 15
observation_state_cb () (tmcproto- OffCommand (class in tmcproto-
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode type.subarraynode.low.src.subarraynode.low.off_command),
method), 14 26
observation_state_cb () (tmcproto- On (class in tmcproto-
type.subarraynode.low.src.subarraynode.low.subarray_node_type.cspmaterleafnode.src.cspmaterleafnode.csp_master_leaf_node.SubarrayNode
method), 24 40
ObsReset (class in tmcproto- On (class in tmcproto-
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.obsreset_command),
66 42
type.cspmaterleafnode.src.cspmaterleafnode.on_command),
ObsReset (class in tmcproto- On (class in tmcproto-
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdpsubarray_LeafNode),
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_command)

58
 On (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode),
 51
 on_cmd_ended_cb () (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.
 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.On
 method), 40
 on_cmd_ended_cb () (tmcproto- type.subarraynode.src.subarraynode.subarray_node.LowSubarrayNode.
 type.cspmasterleafnode.src.cspmasterleafnode.on_command_method), 24
 method), 42
 on_cmd_ended_cb () (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leafnode.MccsMasterLeafNode.OnCommand
 method), 75
 on_cmd_ended_cb () (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.on_command_method), 14
 method), 58
 on_cmd_ended_cb () (tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.On
 method), 51
 OnCommand (class in tmcproto- type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.
 type.subarraynode.src.subarraynode.on_command), method), 14
 15
 OnCommand (class in tmcproto- type.subarraynode.src.subarraynode.subarray_node.LowSubarrayNode.
 type.subarraynode.src.subarraynode.on_command), method), 24
 26
 read_subarray1HealthState () (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
P
 pointing_state_cb () (tmcproto- read_subarray1HealthState () (tmcproto-
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode.
 type.centralnode.src.centralnode.central_node_low.CentralNodeLow.
 method), 14
 method), 11
R
 read_activeProcessingBlocks () (tmcproto- read_subarray2HealthState () (tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leafnode.SdpSubarrayLeafNode.
 method), 49
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
 read_activityMessage () (tmcproto- read_subarray3HealthState () (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
 read_activityMessage () (tmcproto- readyTelescopeHealthState () (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
 read_activityMessage () (tmcproto- readTelescopeHealthState () (tmcproto-
 type.centralnode.src.centralnode.central_node_low.CentralNodeLow.
 method), 11
 type.centralnode.src.centralnode.central_node_low.CentralNodeLow.
 method), 11
 read_activityMessage () (tmcproto- receive_addresses_cb () (tmcproto-
 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leafnode.CspMasterLeafNode
 method), 40
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 method), 14
 read_activityMessage () (tmcproto- receive_addresses
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode (tmcproto-
 method), 35
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
 attribute), 49
 read_activityMessage () (tmcproto- receptorIDList (tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leafnode.MccsMasterLeafNode (tmcproto-
 method), 77
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 attribute), 15
 read_activityMessage () (tmcproto- release_csp_resources () (tmcproto-
 type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_subarray_leafnode.MccsSubarrayLeafNode (tmcproto-
 method), 83
 type.subarraynode.src.subarraynode.release_all_resources_comm
 method), 17

release_sdp_resources () (tmcproto- type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
type.subarraynode.src.subarraynode.release_all_resources_command.ReleaseAllResourcesCommand
method), 17 Restart () (tmcproto-

ReleaseAllResources (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_resources_command),
60 restart_cmd_ended_cb () (tmcproto-

ReleaseAllResources (class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_comm
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode),
51 restart_cmd_ended_cb () (tmcproto-

ReleaseAllResources () (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.SdpSubarrayLeafNode
method), 47 RestartCommand (class in tmcproto-

releaseallresources_cmd_ended_cb () (tm- type.subarraynode.src.subarraynode.restart_command),
cprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.release_resources_command.ReleaseAllResources
method), 61

releaseallresources_cmd_ended_cb () (tm- **S**
cprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode.ReleaseAllResourcesCommand.subarray_node.Subarray
method), 51 attribute), 15

ReleaseAllResourcesCommand Scan (class in tmcproto-
(class in tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.scan_command
type.subarraynode.src.subarraynode.release_all_resources_command),
17 Scan (class in tmcproto-

ReleaseAllResourcesCommand type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.
(class in tmcproto- 56
type.subarraynodelow.src.subarraynodelow.release_all_resources_command.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 26

ReleaseResources (class in tmcproto- Scan () (tmcprototype.mccsubarrayleafnode.src.mccsubarrayleafnode.m
type.centralnode.src.centralnode.release_resources_command),
5 method), 82

ReleaseResources (in module tmcproto- Scan () (tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_
type.centralnode.src.centralnode.central_node), scan_cmd_ended_cb () (tmcproto-
3 method), 48 type.mccsubarrayleafnode.src.mccsubarrayleafnode.mccs_suba

ReleaseResources () (tmcproto- method), 82
type.centralnode.src.centralnode.central_node.CentralNode cmd_ended_cb () (tmcproto-

ReleaseResources () (tmcproto- method), 62
type.centralnodelow.src.centralnodelow.central_node_low.CentralNodeLow cmd_ended_cb () (tmcproto-

ReleaseResources () (tmcproto- method), 56
type.mccsmasterleafnode.src.mccsmasterleafnode.mccsmasterleafnode.MccsMasterLeafNode (class in tmcproto-

releaseresources_cmd_ended_cb () (tmcpro- 18
totype.mccsmasterleafnode.src.mccsmasterleafnode.mccsmasterleafnode.MccsMasterLeafNode.ReleaseResourcesCommand
method), 76 type.subarraynodelow.src.subarraynodelow.scan_command),

remove_receptors_from_group () (tmcproto- 27
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode (tmcprototype.subarraynode.src.subarraynode.subarray_node.Sub
method), 15 attribute), 15

Restart (class in tmcproto- scanID (tmcprototyped.subarraynodelow.src.subarraynodelow.subarray_no
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.restart_command),
65 attribute), 24 SdpMasterLeafNodeFQDN (tmcproto-

Restart (class in tmcproto- type.centralnode.src.centralnode.central_node.CentralNode
type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarrayleafnode),
54 attribute), 54 SdpSubarrayFQDN (tmcproto-

Restart () (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray.

attribute), 48
 SdpSubarrayFQDN (*tmcproto-* StandByTelescope (*in module tmcproto-*
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
attribute), 13
 sdpSubarrayHealthState (*tmcproto-* StandByTelescope (*tmcproto-*
type.sdpSubarrayleafnode.src.sdpSubarrayleafnode.sdp_subarrayleafnode.SdpSubarrayLeafNode
attribute), 49
 SdpSubarrayLeafNode (*class in tmcproto-* StandByTelescope (*tmcproto-*
type.sdpSubarrayleafnode.src.sdpSubarrayleafnode.sdp_subarrayleafnode)
 47
method), 9
 SdpSubarrayLeafNode.InitCommand StartCapture (*tmcproto-*
(class in tmcproto- *type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode*
type.sdpSubarrayleafnode.src.sdpSubarrayleafnode.sdp_subarrayleafnode)
 47
method), 35
 SdpSubarrayLNFQDN (*tmcproto-* *type.centralnode.src.centralnode.central_node*),
type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
attribute), 13
 sdpSubarrayObsState (*tmcproto-* *type.centralnode.src.centralnode.central_node.CentralNode*
type.sdpSubarrayleafnode.src.sdpSubarrayleafnode.sdp_subarrayleafnode.SdpSubarrayLeafNode
attribute), 49
 set_dish_name_number (*tmcproto-* *type.centralnode.src.centralnode.central_node_low.CentralNode*
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35
 set_observer_lat_long_alt (*tmcproto-* *type.subarraynode.src.subarraynode.end_command.EndCommand*
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 35
 SetOperateMode (*tmcproto-* *type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode*
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 31
 SetStandbyFPMode (*tmcproto-* *type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode*
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 31
 SetStandbyLPMode (*tmcproto-* *type.centralnode.src.centralnode.stow_antennas_command*),
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 32
 SetStowMode (*tmcproto-* *type.centralnode.src.centralnode.central_node*),
type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
method), 32
 Slew (*tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode*
method), 32
 Standby (*class in tmcproto-* subarray1HealthState (*tmcproto-*
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
 41
attribute), 3
 Standby (*class in tmcproto-* subarray1HealthState (*tmcproto-*
type.cspmasterleafnode.src.cspmasterleafnode.standby_command)
 42
attribute), 11
 Standby (*tmcproto-* subarray2HealthState (*tmcproto-*
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
method), 39
attribute), 3
 standby_cmd_ended_cb (*tmcproto-* subarray3HealthState (*tmcproto-*
type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.Standby
method), 41
attribute), 3
 standby_cmd_ended_cb (*tmcproto-* SubarrayNode (*class in tmcproto-*
type.cspmasterleafnode.src.cspmasterleafnode.standby_command)
 41
attribute), 3

13		tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode
SubarrayNode	(class in tmcprototype.subarraynode)	(module), 64
23		tmcprototype.subarraynode
SubarrayNode.InitCommand	(class in tmcprototype.subarraynode)	(module), 66
13		tmcprototype.subarraynode
SubarrayNode.InitCommand	(class in tmcprototype.subarraynode)	(module), 59
23		tmcprototype.subarraynode
T		
telescopeHealthState	(tmcprototype.centralnode)	(module), 65
attribute), 3		
telescopeHealthState	(tmcprototype.centralnode)	(module), 62
attribute), 11		
TMAAlarmHandler	(tmcprototype.subarraynode)	(module), 19
attribute), 2		
TMAAlarmHandler	(tmcprototype.subarraynode)	(module), 16
attribute), 10		
tmcprototype.centralnode.src.centralnode		(module), 19
(module), 4		
tmcprototype.centralnode.src.centralnode		(module), 19
(module), 1		
tmcprototype.centralnode.src.centralnode		(module), 20
(module), 5		
tmcprototype.centralnode.src.centralnode		(module), 15
(module), 6		
tmcprototype.centralnode.src.centralnode		(module), 15
(module), 7		
tmcprototype.cspmasterleafnode.src.cspmasterleafnode		(module), 17
(module), 39		
tmcprototype.cspmasterleafnode.src.cspmasterleafnode		(module), 20
(module), 41		
tmcprototype.cspmasterleafnode.src.cspmasterleafnode		(module), 18
(module), 42		
tmcprototype.cspmasterleafnode.src.cspmasterleafnode		(module), 13
(module), 42		
tmcprototype.dishleafnode.src.dishleafnode		(module), 20
(module), 29		
tmcprototype.mccsmasterleafnode.src.mccsmasterleafnode		(module), 25
(module), 73		
tmcprototype.mccssubarrayleafnode.src.mccssubarrayleafnode		(module), 25
(module), 79		
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode		(module), 25
(module), 64		
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode		(module), 26
(module), 59		
tmcprototype.sdpsubarrayleafnode.src.sdpsubarrayleafnode		(module), 26
(module), 61		

tmcprototype.subarraynodelow.src.subarraynodelow.track_command
 (module), 26 write_activityMessage() (tmcproto-
 tmcprototype.subarraynodelow.src.subarraynodelow.track_command.src.subarraynodelow.subarray_node.SubarrayNode
 (module), 26 method), 15
 tmcprototype.subarraynodelow.src.subarraynodelow.write_activityMessage() (tmcproto-
 (module), 27 type.subarraynodelow.src.subarraynodelow.subarray_node_low.S
 tmcprototype.subarraynodelow.src.subarraynodelow.write_activityMessage() (tmcproto-
 (module), 23 type.subarraynodelow.src.subarraynodelow.subarray_node_low.S
 TMLowSubarrayNodes (tmcproto- type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray
 type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
 attribute), 10 method), 40
 TMMidSubarrayNodes (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 attribute), 2
 Track() (tmcprototype.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 method), 33
 Track() (tmcprototype.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 method), 13
 track_thread() (tmcproto-
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 method), 35
 TrackCommand (class in tmcproto-
 type.subarraynode.src.subarraynode.track_command),
 20

V

validate_obs_state() (tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode
 method), 49
 validate_obs_state() (tmcproto-
 type.subarraynode.src.subarraynode.subarray_node.SubarrayNode
 method), 15

W

write_activityMessage() (tmcproto-
 type.centralnode.src.centralnode.central_node.CentralNode
 method), 3
 write_activityMessage() (tmcproto-
 type.centralnodelow.src.centralnodelow.central_node_low.CentralNode
 method), 11
 write_activityMessage() (tmcproto-
 type.cspmasterleafnode.src.cspmasterleafnode.csp_master_leaf_node.CspMasterLeafNode
 method), 40
 write_activityMessage() (tmcproto-
 type.dishleafnode.src.dishleafnode.dish_leaf_node.DishLeafNode
 method), 35
 write_activityMessage() (tmcproto-
 type.mccsmasterleafnode.src.mccsmasterleafnode.mccs_master_leaf_node.MccsMasterLeafNode
 method), 77
 write_activityMessage() (tmcproto-
 type.mccssubarrayleafnode.src.mccssubarrayleafnode.mccs_subarray_leaf_node.MccsSubarrayLeafNode
 method), 83
 write_activityMessage() (tmcproto-
 type.sdpsubarrayleafnode.src.sdpsubarrayleafnode.sdp_subarray_leaf_node.SdpSubarrayLeafNode