TMC Leaf Nodes Documentation

Release 1.0

NCRA India

Mar 17, 2022

GETTING STARTED

This project is developing the TMC Leaf Nodes component of the Telescope Monitoring and Control (TMC) prototype, for the Square Kilometre Array.

This page contains instructions for software developers who want to get started with usage and development of the TMC Leaf Nodes.

CHAPTER

ONE

BACKGROUND

Detailed information on how the SKA Software development community works is available at the SKA software developer portal. There you will find guidelines, policies, standards and a range of other documentation.

CHAPTER

TWO

SET UP YOUR DEVELOPMENT ENVIRONMENT

This project is structured to use k8s for development and testing so that the build environment, test environment and test results are all completely reproducible and are independent of host environment. It uses `make` to provide a consistent UI (run `make help` for targets documentation).

2.1 Install minikube

You will need to install *minikube* or equivalent k8s installation in order to set up your test environment. You can follow the instruction here: :: git clone git@gitlab.com:ska-telescope/sdi/deploy-minikube.git cd deploy-minikube make all eval $(minikube docker-env)

Please note that the command `eval $(minikube docker-env)` will point your local docker client at the docker-in-docker for minikube. Use this only for building the docker image and another shell for other work.

2.2 How to Use

Clone this repo: :: git clone https://gitlab.com/ska-telescope/ska-tmc-leafnodes.git cd ska-tmc-leafnodes

Install dependencies :: apt update apt install -y curl git build-essential libboost-python-dev libtango-dev curl -sSL https://raw.githubusercontent.com/python-poetry/poetry/master/get-poetry.py | python3 - source $HOME/.poetry/env

Please note that:

- the *libtango-dev* will install an old version of the TANGO-controls framework (9.2.5);
- the best way to get the framework is compiling it (instructions can be found here);
- the above script has been tested with Ubuntu 20.04.

During this step, `libtango-dev` instalation can ask for the Tango Server IP:PORT. Just accept the default proposed value.

Install python requirements for linting and unit testing: :: $ poetry install

Activate the poetry environment: :: $ source $(poetry env info –path)/bin/activate

Alternate way to install and activate poetry

Follow the steps till installation of dependencies then run below command: :: $ virtualenv cn_venv $ source cn_venv/bin/activate $ make requirements

Run python-test: :: $ make python-test PyTango 9.3.3 (9, 3, 3) PyTango compiled with: Python : 3.8.5 Numpy : 0.0.0 ## output generated from a WSL windows machine Tango : 9.2.5 Boost : 1.71.0

PyTango runtime is: Python : 3.8.5 Numpy : None Tango : 9.2.5

PyTango running on: uname_result(system='Linux', node='LAPTOP-5LBGJH83', release='4.19.128-microsoft-standard', version='#1 SMP Tue Jun 23 12:58:10 UTC 2020', machine='x86_64', processor='x86_64')

============================= test session starts ============================== platform linux – Python 3.8.5, pytest-5.4.3, py-1.10.0, pluggy-0.13.1 – /home/ [....]

———————————— JSON report ————————————- JSON report written to: build/reports/report.json (165946 bytes)

———— coverage: platform linux, python 3.8.5-final-0 ———— Coverage HTML written to dir build/htmlcov Coverage XML written to file build/reports/code-coverage.xml

======================== 48 passed, 5 deselected in 42.42s =======================

Formatting the code: :: $ make python-format [...] ———————————————————————————— Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)

Python linting: :: $ make python-lint [...] ———————————————————————————— Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)

CHAPTER

THREE

TMC LEAF NODES CODE QUALITY GUIDELINES

3.1 Code formatting / style

3.1.1 Black

TMC Leaf Nodes uses the `black` code formatter to format its code. Formatting can be checked using the command `make python-format`.

The CI pipeline does check that if code has been formatted using black or not.

3.1.2 Linting

TMC Leaf Nodes uses below libraries/utilities for linting. Linting can be checked using command `make python-lint`.

- **`isort` - It provides a command line utility, Python library and** plugins for various editors to quickly sort all your imports.
- `black` - It is used to check if the code has been blacked.
- **`flake8` - It is used to check code base against coding style (PEP8),** programming errors (like "library imported but unused" and "Undefined name"),etc.
- **`pylint` - It is looks for programming errors, helps enforcing a coding standard,** sniffs for code smells and offers simple refactoring suggestions.

3.2 Test coverage

TMC Leaf Nodes uses pytest to test its code, with the pytest-cov plugin for measuring coverage.

CHAPTER

FOUR

SKA_TMC_SDPMASTERLEAFNODE PACKAGE

4.1 Subpackages

4.1.1 ska_tmc_sdpmasterleafnode.commands package

Submodules

ska_tmc_sdpmasterleafnode.commands.abstract_command module

class ska_tmc_sdpmasterleafnode.commands.abstract_command.**SdpMLNCommand**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

Abstract command class for all SdpMasterLeafNode

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

check_unresponsive()

do(*argin=None*)

init_adapter()

ska_tmc_sdpmasterleafnode.commands.disable_command module

Disable command class for SDPMasterLeafNode.

class ska_tmc_sdpmasterleafnode.commands.disable_command.**Disable**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Disable() command.

Disable command on SdpMasterLeafNode invokes disable command on Sdp Master device.

do(*argin=None*)

Method to invoke Disable command on Sdp Master.

ska_tmc_sdpmasterleafnode.commands.off_command module

Off command class for SDPMasterLeafNode.

class ska_tmc_sdpmasterleafnode.commands.off_command.**Off**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Off() command.

Off command on SdpMasterLeafNode enables the telescope to perform further operations and observations. It Invokes Off command on Sdp Master device.

do(*argin=None*)
Method to invoke Off command on Sdp Master.

ska_tmc_sdpmasterleafnode.commands.on_command module

On command class for SDPMasterLeafNode.

class ska_tmc_sdpmasterleafnode.commands.on_command.**On**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's On() command.

On command on SdpmasterLeafNode enables the telescope to perform further operations and observations. It Invokes On command on Sdp Master device.

do(*argin=None*)
Method to invoke On command on Sdp Master.

ska_tmc_sdpmasterleafnode.commands.standby_command module

Standby command class for SDPMasterLeafNode.

class ska_tmc_sdpmasterleafnode.commands.standby_command.**Standby**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Standby() command.

Standby command on SdpMasterLeafNode invokes Standby command on Sdp Master device.

do(*argin=None*)
Method to invoke Standby command on Sdp Master.

Module contents

class ska_tmc_sdpmasterleafnode.commands.**Disable**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Disable() command.

Disable command on SdpMasterLeafNode invokes disable command on Sdp Master device.

do(*argin=None*)
Method to invoke Disable command on Sdp Master.

class ska_tmc_sdpmasterleafnode.commands.**Off**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Off() command.

Off command on SdpMasterLeafNode enables the telescope to perform further operations and observations. It Invokes Off command on Sdp Master device.

do(*argin=None*)
Method to invoke Off command on Sdp Master.

class ska_tmc_sdpmasterleafnode.commands.**On**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's On() command.

On command on SdpmasterLeafNode enables the telescope to perform further operations and observations. It Invokes On command on Sdp Master device.

do(*argin=None*)
Method to invoke On command on Sdp Master.

class ska_tmc_sdpmasterleafnode.commands.**Standby**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpMasterLeafNode's Standby() command.

Standby command on SdpMasterLeafNode invokes Standby command on Sdp Master device.

do(*argin=None*)
Method to invoke Standby command on Sdp Master.

4.1.2 ska_tmc_sdpmasterleafnode.manager package

Submodules

ska_tmc_sdpmasterleafnode.manager.component_manager module

This module implements ComponentManager class for the Sdp Master Leaf Node.

class ska_tmc_sdpmasterleafnode.manager.component_manager.**SdpMLNComponentManager**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_component_manager.ska_tmc_common.tmc_component_manager.TmcLeafNodeComponentManager._name

A component manager for The SDP Master Leaf Node component.

It supports:

- Monitoring its component, e.g. detect that it has been turned off or on
- Controlling the behaviour of SDP Master.

device_failed(*exception*)
Set a device to failed and call the relative callback if available

Parameters **exception** – an exception

Type Exception

update_device_info(*sdp_master_dev_name*)

Module contents

class ska_tmc_sdpmasterleafnode.manager.**SdpMLNComponentManager**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_component_manager.ska_tmc_common.tmc_component_manager.TmcLeafNodeComponentManager._name

A component manager for The SDP Master Leaf Node component.

It supports:

- Monitoring its component, e.g. detect that it has been turned off or on
- Controlling the behaviour of SDP Master.

device_failed(*exception*)
Set a device to failed and call the relative callback if available

Parameters **exception** – an exception

Type Exception

update_device_info(*sdp_master_dev_name*)

4.2 Submodules

4.3 ska_tmc_sdpmasterleafnode._sdp_master_leaf_node module

SDP Master Leaf node acts as a SDP contact point for Master Node and also to monitor and issue commands to the SDP Master.

class ska_tmc_sdpmasterleafnode.sdp_master_leaf_node.**SdpMasterLeafNode**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_base.ska_tmc_base.SKABaseDevice._name

SDP Master Leaf node acts as a SDP contact point for Master Node and also to monitor and issue commands to the SDP Master.

class InitCommand(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_base.SKABaseDevice.ska_tmc_base.SKABaseDevice.InitCommand._name

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

do()
Initializes the attributes and properties of the SdpMasterLeafNode.

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

rtype: (ResultCode, str)

Off()
This command invokes Off() command on Sdp Master.

always_executed_hook()

create_component_manager()

delete_device()

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

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

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

Return type boolean

is_Off_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_On_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_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

read_commandExecuted()
Return the commandExecuted attribute.

read_sdpMasterDevName()
Return the sdpmasterdevname attribute.

write_sdpMasterDevName(*value*)
Set the sdpmasterdevname attribute.

ska_tmc_sdpmasterleafnode.sdp_master_leaf_node.**main**(*args=None*, ***kwargs*)
Runs the SdpMasterLeafNodeMid. :param args: Arguments internal to TANGO

Parameters **kwargs** – Arguments internal to TANGO

Returns SdpMasterLeafNodeMid TANGO object.

4.4 Module contents

CHAPTER

FIVE

SKA_TMC_SDPSUBARRAYLEAFNODE PACKAGE

5.1 Subpackages

5.1.1 ska_tmc_sdpsubarrayleafnode.commands package

Submodules

ska_tmc_sdpsubarrayleafnode.commands.abstract_command module

class ska_tmc_sdpsubarrayleafnode.commands.abstract_command.**AbstractOnOff**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

class ska_tmc_sdpsubarrayleafnode.commands.abstract_command.**AbstractRestartObsReset**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

class ska_tmc_sdpsubarrayleafnode.commands.abstract_command.**AbstractScanEnd**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

class ska_tmc_sdpsubarrayleafnode.commands.abstract_command.**SdpSLNCommand**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

check_allowed()

check_op_state(*command_name*)

check_unresponsive()

init_adapter()

ska_tmc_sdpsubarrayleafnode.commands.off_command module

Off command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.off_command.**Off**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpsubarrayLeafNode's Off() command.

Off command on SdpsubarrayLeafNode enables the telescope to perform further operations and observations. It Invokes Off command on Sdp Subarray device.

do(*argin=None*)

Method to invoke Telescope Off command on Sdp Subarray.

ska_tmc_sdpsubarrayleafnode.commands.on_command module

On command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.on_command.**On**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpsubarrayLeafNode's On() command.

On command on SdpsubarrayLeafNode enables the telescope to perform further operations and observations. It Invokes On command on Sdp Subarray device.

do(*argin=None*)

Method to invoke Telescope On command on Sdp Subarray.

ska_tmc_sdpsubarrayleafnode.commands.assign_resources_command module

AssignResouces command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.assign_resources_command.**AssignResources**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarayLeafNode's AssignResources() command.

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.

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin*)

Method to invoke AssignResources command on SDP Subarray.

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

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

eb_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

{“interface”:”https://schema.skao.int/ska-sdp-assignres/0.3”, “eb_id”:”eb-mvp01-20200325-00001”,”max_length”:100.0,”scan_types”:[{“scan_type_id”:”science_A”, “reference_frame”:”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]]}]} ,{“scan_type_id”:”calibration_B”,”reference_frame”:”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”:[{“pb_id”:”pb-mvp01-20200325-00001”,”workflow”: {“kind”:”realtime”,”name”:”vis_receive”,”version”:”0.1.0”},”parameters”:{}},{“pb_id”: “pb-mvp01-20200325-00002”,”workflow”:{“kind”:”realtime”,”name”:”test_realtime”,”version”:”0.1.0”}, “parameters”:{}},{“pb_id”:”pb-mvp01-20200325-00003”,”workflow”:{“kind”:”batch”,”name”:”ical”, “version”:”0.1.0”},”parameters”:{},”dependencies”:[{“pb_id”:”pb-mvp01-20200325-00001”, “kind”:[“visibilities”]}]},{“pb_id”:”pb-mvp01-20200325-00004”,”workflow”:{“kind”:”batch”,”name”:”dpreb”, “version”:”0.1.0”},”parameters”:{},”dependencies”:[{“pb_id”:”pb-mvp01-20200325-00003”,”kind”: [“calibration”]}]}]}

Note: Enter input without spaces

Returns None

ska_tmc_sdpsubarrayleafnode.commands.configure_command module

Configure command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.configure_command.**Configure**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode’s Configure() command.

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

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin*)
Method to invoke Configure command on SDP Subarray.

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

Example

{ "interface": "https://schema.skao.int/ska-sdp-configure/0.3", "scan_type": "science_A" }

Returns None

ska_tmc_sdpsubarrayleafnode.commands.end_command module

End command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.end_command.**End**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's End() command.

Invokes End command on SDP Subarray to end the current Scheduling Block.

do(*argin=None*)
Method to invoke End command on SDP Subarray.

Parameters **argin** – None

Returns None

ska_tmc_sdpsubarrayleafnode.commands.endscan_command module

EndScan command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.endscan_command.**EndScan**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's EndScan() command.

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

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke EndScan command on SDP Subarray.

ska_tmc_sdpsubarrayleafnode.commands.obsreset_command module

ObsReset command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.obsreset_command.**ObsReset**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's ObsReset() command.

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

do(*argin=None*)

Method to invoke ObsReset command on SDP Subarray.

Parameters **argin** – None

Returns None

ska_tmc_sdpsubarrayleafnode.commands.release_resources_command module

ReleaseResources command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.release_resources_command.**ReleaseResources**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarayLeafNode's ReleaseResources() command.

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

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)

Method to invoke ReleaseResources command on SDP Subarray.

Parameters **argin** – None.

Returns None

ska_tmc_sdpsubarrayleafnode.commands.reset_command module

Reset command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.reset_command.**Reset**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Reset() command.

Command to reset the SDP Subarray and bring it to its initial state.

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke Reset command on SDP Subarray.

Parameters **argin** – None

Returns None

ska_tmc_sdpsubarrayleafnode.commands.restart_command module

Restart command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.restart_command.**Restart**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Restart() command.

Command to reset the SDP Subarray and bring it to its RESTARTING state.

do(*argin=None*)
Method to invoke Restart command on SDP Subarray.

Parameters **argin** – None

Returns None

ska_tmc_sdpsubarrayleafnode.commands.scan_command module

Scan command class for SDPSubarrayLeafNode.

class ska_tmc_sdpsubarrayleafnode.commands.scan_command.**Scan**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Scan() command.

Invoke Scan command to SDP Subarray.

do(*argin*)
Method to invoke Scan command on SDP Subarray.

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

Example

{ "interface": "https://schema.skao.int/ska-sdp-scan/0.3", "scan_id": 1

}

Returns None

Module contents

class ska_tmc_sdpsubarrayleafnode.commands.**Abort**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Abort() command.

Command to abort the SDP Subarray and bring it to its ABORTED state.

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke Abort command on SDP Subarray.

Parameters **argin** – None

Returns None

Raises **Exception if error occurs while invoking command on SDP Subarray.** –

class ska_tmc_sdpsubarrayleafnode.commands.**AssignResources**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarayLeafNode's AssignResources() command.

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.

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin*)
Method to invoke AssignResources command on SDP Subarray.

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

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

eb_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

{“interface”:”https://schema.skao.int/ska-sdp-assignres/0.3”, “eb_id”:”eb-mvp01-20200325-00001”,”max_length”:100.0,”scan_types”:[{“scan_type_id”:”science_A”, “reference_frame”:”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]]}]} ,{“scan_type_id”:”calibration_B”,”reference_frame”:”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”:[{“pb_id”:”pb-mvp01-20200325-00001”,”workflow”: {“kind”:”realtime”,”name”:”vis_receive”,”version”:”0.1.0”},”parameters”:{}},{“pb_id”: “pb-mvp01-20200325-00002”,”workflow”:{“kind”:”realtime”,”name”:”test_realtime”,”version”:”0.1.0”}, “parameters”:{}},{“pb_id”:”pb-mvp01-20200325-00003”,”workflow”:{“kind”:”batch”,”name”:”ical”, “version”:”0.1.0”},”parameters”:{},”dependencies”:[{“pb_id”:”pb-mvp01-20200325-00001”, “kind”:[“visibilities”]}]},{“pb_id”:”pb-mvp01-20200325-00004”,”workflow”:{“kind”:”batch”,”name”:”dpreb”, “version”:”0.1.0”},”parameters”:{},”dependencies”:[{“pb_id”:”pb-mvp01-20200325-00003”,”kind”: [“calibration”]}]}]}

Note: Enter input without spaces

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**Configure**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode’s Configure() command.

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

check_allowed()

Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin*)

Method to invoke Configure command on SDP Subarray.

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

Example

{ “interface”: “https://schema.skao.int/ska-sdp-configure/0.3”, “scan_type”: “science_A” }

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**End**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode’s End() command.

Invokes End command on SDP Subarray to end the current Scheduling Block.

do(*argin=None*)

Method to invoke End command on SDP Subarray.

Parameters **argin** – None

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**EndScan**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's EndScan() command.

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

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke EndScan command on SDP Subarray.

class ska_tmc_sdpsubarrayleafnode.commands.**ObsReset**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's ObsReset() command.

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

do(*argin=None*)
Method to invoke ObsReset command on SDP Subarray.

Parameters **argin** – None

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**Off**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpsubarrayLeafNode's Off() command.

Off command on SdpsubarrayLeafNode enables the telescope to perform further operations and observations. It Invokes Off command on Sdp Subarray device.

do(*argin=None*)
Method to invoke Telescope Off command on Sdp Subarray.

class ska_tmc_sdpsubarrayleafnode.commands.**On**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpsubarrayLeafNode's On() command.

On command on SdpsubarrayLeafNode enables the telescope to perform further operations and observations. It Invokes On command on Sdp Subarray device.

do(*argin=None*)
Method to invoke Telescope On command on Sdp Subarray.

class ska_tmc_sdpsubarrayleafnode.commands.**ReleaseResources**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarayLeafNode's ReleaseResources() command.

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

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke ReleaseResources command on SDP Subarray.

Parameters **argin** – None.

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**Reset**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Reset() command.

Command to reset the SDP Subarray and bring it to its initial state.

check_allowed()
Checks whether this command is allowed It checks that the device is in the right state to execute this command and that all the component needed for the operation are not unresponsive

Returns True if this command is allowed

Return type boolean

do(*argin=None*)
Method to invoke Reset command on SDP Subarray.

Parameters **argin** – None

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**Restart**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Restart() command.

Command to reset the SDP Subarray and bring it to its RESTARTING state.

do(*argin=None*)
Method to invoke Restart command on SDP Subarray.

Parameters **argin** – None

Returns None

class ska_tmc_sdpsubarrayleafnode.commands.**Scan**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_command.ska_tmc_common.tmc_command.TmcLeafNodeCommand._name

A class for SdpSubarrayLeafNode's Scan() command.

Invoke Scan command to SDP Subarray.

do(*argin*)
Method to invoke Scan command on SDP Subarray.

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

Example

{ “interface”: “https://schema.skao.int/ska-sdp-scan/0.3”, “scan_id”: 1

}

Returns None

5.1.2 ska_tmc_sdpsubarrayleafnode.manager package

Submodules

ska_tmc_sdpsubarrayleafnode.manager.component_manager module

This module provided a reference implementation of a BaseComponentManager.

It is provided for explanatory purposes, and to support testing of this package.

class ska_tmc_sdpsubarrayleafnode.manager.component_manager.**SdpSLNComponentManager**(**args: Any*, ***kwargs: Any*)

Bases: ska_tmc_common.tmc_component_manager.ska_tmc_common.tmc_component_manager.TmcLeafNodeComponentManager._name

A component manager for The SDP Subarray Leaf Node component.

It supports:

- Monitoring its component, e.g. detect that it has been turned off or on

device_failed(*exception*)
Return the list of the checked monitored devices

Returns list of the checked monitored devices

get_device()
Return the device info our of the monitoring loop with name dev_name

Parameters **None** –

Returns a device info

Return type DeviceInfo

stop()

update_device_info(*sdp_subarray_dev_name*)

update_device_obs_state(*obs_state*)
Update a monitored device obs state, and call the relative callbacks if available

Parameters

- **dev_name** (*str*) – name of the device
- **obs_state** (*ObsState*) – obs state of the device

update_event_failure()

update_input_parameter()

Module contents

class ska_tmc_sdpsubarrayleafnode.manager.**SdpSLNComponentManager**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.tmc_component_manager.ska_tmc_common.tmc_component_manager.TmcLeafNodeComponentManager._name

A component manager for The SDP Subarray Leaf Node component.

It supports:

- Monitoring its component, e.g. detect that it has been turned off or on

device_failed(*exception*)
Return the list of the checked monitored devices

Returns list of the checked monitored devices

get_device()
Return the device info our of the monitoring loop with name dev_name

Parameters **None** –

Returns a device info

Return type DeviceInfo

stop()

update_device_info(*sdp_subarray_dev_name*)

update_device_obs_state(*obs_state*)
Update a monitored device obs state, and call the relative callbacks if available

Parameters

- **dev_name** (*str*) – name of the device
- **obs_state** (*ObsState*) – obs state of the device

update_event_failure()

update_input_parameter()

class ska_tmc_sdpsubarrayleafnode.manager.**SdpSLNEventReceiver**(**args: Any*, ***kwargs: Any*)
Bases: ska_tmc_common.event_receiver.ska_tmc_common.event_receiver.EventReceiver._name

The SdpSLNEventReceiver class has the responsibility to receive events from the sub devices managed by the Sdp Subarray Leaf Node.

The ComponentManager uses the handle events methods for the attribute of interest. For each of them a callback is defined.

handle_obs_state_event(*evt*)

run()

subscribe_events(*devInfo*)

5.2 Submodules

5.3 ska_tmc_sdpsubarrayleafnode._sdp_subarray_leaf_node module

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 ska_tmc_sdpsubarrayleafnode.sdp_subarray_leaf_node.**SdpSubarrayLeafNode**(**args: Any*, ***kwargs: Any*)

Bases: ska_tango_base.ska_tango_base.SKABaseDevice._name

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

class InitCommand(**args: Any*, ***kwargs: Any*)

Bases: ska_tango_base.SKABaseDevice.ska_tango_base.SKABaseDevice.InitCommand._name

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.

rtype: (ResultCode, str)

Off()

This command invokes Off() command on Sdp Subarray.

always_executed_hook()

create_component_manager()

delete_device()

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

rtype: 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

rtype: boolean

`is_EndScan_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.

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.

rtype: 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

rtype: boolean

`is_Off_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_On_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

`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

rtype: 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.

rtype: boolean

read_commandExecuted()
Return the commandExecuted attribute.

read_lastDeviceInfoChanged()

read_sdpSubarrayDevName()
Return the sdpsubarraydevname attribute.

update_device_callback(*devInfo*)

write_sdpSubarrayDevName(*value*)
Set the sdpsubarraydevname attribute.

ska_tmc_sdpsubarrayleafnode.sdp_subarray_leaf_node.**main**(*args=None*, ***kwargs*)
Runs the SdpSubarrayLeafNode Tango device. :param args: Arguments internal to TANGO

Parameters **kwargs** – Arguments internal to TANGO

Returns integer. Exit code of the run method.

5.4 Module contents

SdpSubarrayLeafNode

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search
- search

PYTHON MODULE INDEX

S

INDEX

E

G

H

I

M

O

R

S

U

W