Reference Manual

Generated by Doxygen 1.8.6

Tue Feb 16 2016 02:38:37

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

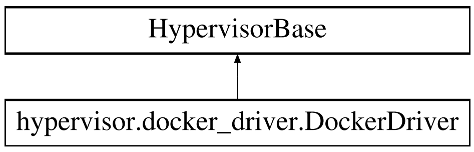
Chapter 3

Class Documentation

3.1 hypervisor.docker_driver.DockerDriver Class Reference

docker driver for nfio.

Inheritance diagram for hypervisor.docker_driver.DockerDriver:



Public Member Functions

- def **__init__**
- def get_id

    Returns a container's ID.
- def get_ip

    Returns a container's IP address.
- def deploy

    Deploys a docker container.
- def start

    Starts a docker container.
- def restart

    Restarts a docker container.
- def stop

    Stops a docker container.
- def pause

    Pauses a docker container.
- def unpause

    Unpauses a docker container.
- def destroy

    Destroys a docker container.
- def execute_in_guest

    Executed commands inside a docker container.
- def guest_status

    Returns the status of a docker container.

3.1.1 Detailed Description

docker driver for nfio.

This class provides methods for managing docker containers.

3.1.2 Member Function Documentation

3.1.2.1 def hypervisor.docker_driver.DockerDriver.deploy (*self, host, user, image_name, vnf_name, is_privileged =* `True`)

Deploys a docker container.

Parameters

host	IP address or hostname of the machine where the docker container is to be deployed
user	name of the user who owns the VNF
image_name	docker image name for the VNF
vnf_name	name of the VNF instance
is_privileged	if True then the container is deployed in privileged mode

Returns

docker container ID

3.1.2.2 def hypervisor.docker_driver.DockerDriver.destroy (*self, host, user, vnf_name, force =* `True`)

Destroys a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF
force	if set to False then a running VNF will not be destroyed. default is True

3.1.2.3 def hypervisor.docker_driver.DockerDriver.execute_in_guest (*self, host, user, vnf_name, cmd*)

Executed commands inside a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF
cmd	the command to execute inside the container

Returns

The output of the command passes as cmd

3.1.2.4 def hypervisor.docker_driver.DockerDriver.get_id (*self, host, user, vnf_name*)

Returns a container's ID.

Parameters

host	IP address or hostname of the machine where the docker container is deployed
user	name of the user who owns the VNF
vnf_name	name of the VNF instance whose ID is being queried

Returns

docker container ID.

3.1.2.5 def hypervisor.docker_driver.DockerDriver.get_ip (*self*, *host*, *user*, *vnf_name*)

Returns a container's IP address.

Parameters

host	IP address or hostname of the machine where the docker container is deployed
user	name of the user who owns the VNF
vnf_name	name of the VNF instance whose ID is being queried

Returns

docker container's IP.

3.1.2.6 def hypervisor.docker_driver.DockerDriver.guest_status (*self*, *host*, *user*, *vnf_name*)

Returns the status of a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF

Returns

current state of the docker container

3.1.2.7 def hypervisor.docker_driver.DockerDriver.pause (*self*, *host*, *user*, *vnf_name*)

Pauses a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF

3.1.2.8 def hypervisor.docker_driver.DockerDriver.restart (*self*, *host*, *user*, *vnf_name*)

Restarts a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF

3.1.2.9 def hypervisor.docker_driver.DockerDriver.start (*self, host, user, vnf_name, is_privileged* = `True`)

Starts a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF
is_privileged	if True then the container is started in privileged mode

3.1.2.10 def hypervisor.docker_driver.DockerDriver.stop (*self, host, user, vnf_name*)

Stops a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF

3.1.2.11 def hypervisor.docker_driver.DockerDriver.unpause (*self, host, user, vnf_name*)

Unpauses a docker container.

Parameters

host	IP address or hostname of the machine/VM where the docker container is deployed
user	name of the user
vnf_name	name of the VNF

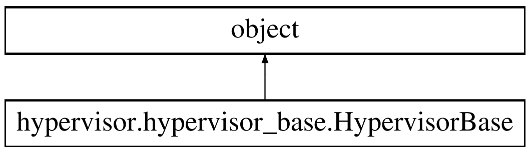
The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/hypervisor/docker_driver.py

3.2 hypervisor.hypervisor_base.HypervisorBase Class Reference

Base class for hypervisors.

Inheritance diagram for hypervisor.hypervisor_base.HypervisorBase:



Public Member Functions

- def get_id

  Returns the hypervisor specific ID of the VM or container.
- def deploy

  Deploys a VM or continer.
- def pause

  Pauses a VM or continer.
- def destroy

  Destroys a VM or continer.
- def execute_in_guest

  Executes a command in the VM or continer.
- def guest_status

  Returns the current status of a VM or continer.

3.2.1 Detailed Description

Base class for hypervisors.

This class must be extended by a hypervisor driver.

3.2.2 Member Function Documentation

3.2.2.1 def hypervisor.hypervisor_base.HypervisorBase.deploy (*self*)

Deploys a VM or continer.

Args: Defined in derived class.

Returns: Hypervisor specific return code.

3.2.2.2 def hypervisor.hypervisor_base.HypervisorBase.destroy (*self*)

Destroys a VM or continer.

Args: Defined in derived class.

Returns: Hypervisor specific return code.

3.2.2.3 def hypervisor.hypervisor_base.HypervisorBase.execute_in_guest (*self*)

Executes a command in the VM or continer.

Args: Defined in derived class.

Returns: Hypervisor specific return code.

3.2.2.4 def hypervisor.hypervisor_base.HypervisorBase.get_id (*self*)

Returns the hypervisor specific ID of the VM or container.

Args: Defined in derived class.

Returns: Hypervisor specific ID for a VM or container.

3.2.2.5 def hypervisor.hypervisor_base.HypervisorBase.guest_status (*self*)

Returns the current status of a VM or continer.

Args: Defined in derived class.

Returns: Current status of a VM or container.

3.2.2.6 def hypervisor.hypervisor_base.HypervisorBase.pause (*self*)

Pauses a VM or continer.

Args: Defined in derived class.

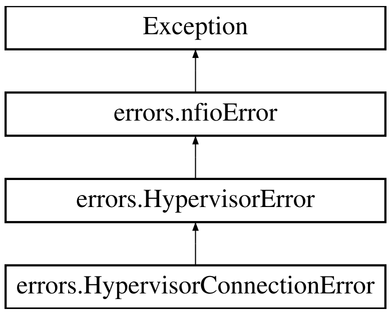
Returns: Hypervisor specific return code.

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/hypervisor/hypervisor_base.py

3.3 errors.HypervisorConnectionError Class Reference

Inheritance diagram for errors.HypervisorConnectionError:



Public Member Functions

- def **__init__**

Public Attributes

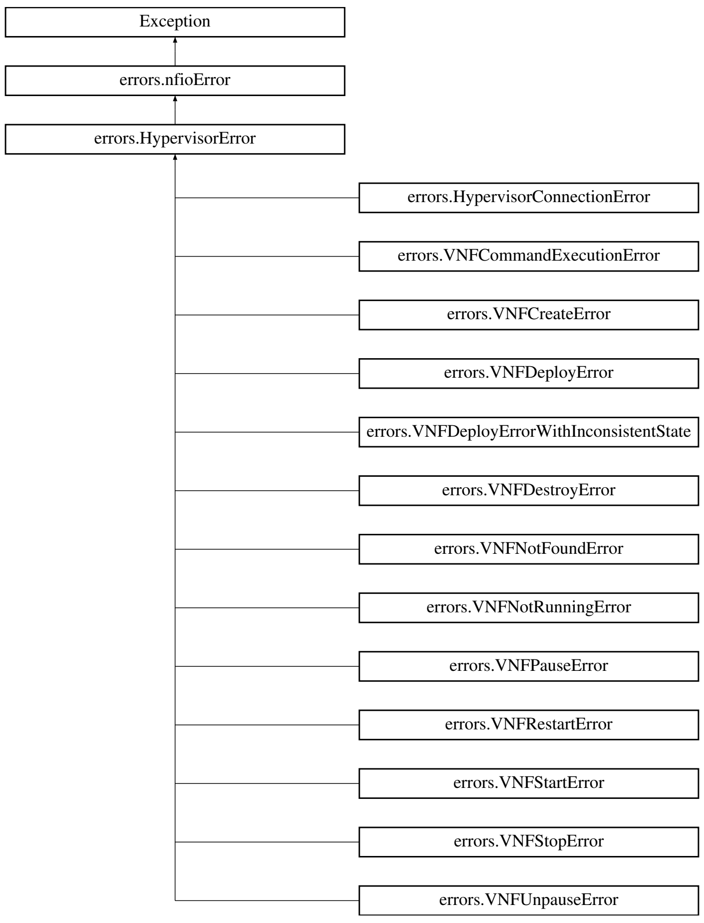
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.4 errors.HypervisorError Class Reference

Inheritance diagram for errors.HypervisorError:



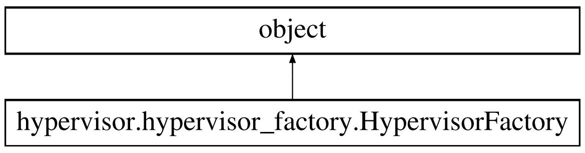
The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.5 hypervisor.hypervisor_factory.HypervisorFactory Class Reference

A singletone class for creating hypervisor driver objects.

Inheritance diagram for hypervisor.hypervisor_factory.HypervisorFactory:



Public Member Functions

- def __init__

    Instantiates a HypervisorFactory object.

Static Public Member Functions

- def get_hypervisor_instance

Returns the hypervisor driver nstance.

3.5.1 Detailed Description

A singletone class for creating hypervisor driver objects.

For an instantiation of nf.io there can be exactly one object of only one type of hyperviosr. HyervisorFactory takes care of the creation logic.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 def hypervisor.hypervisor_factory.HypervisorFactory.__init__ (*self*, *hypervisor_type* = "DockerDriver")

Instantiates a HypervisorFactory object.

Args: hypervisor_type: The type of hypervisor object to instantiate. Valid hypervisor types are 'DockerDriver' and 'Libvirt' for the time being.

Returns: Nothing. Initializaes the factory object.

Note: If this factory class is instantiated multiple times with different types of hypervisor_type argument then it raises a ValueError.

If this factory class is instantiated with a hypervisor type other than Docker or Libvirt it raises a TypeError.

3.5.3 Member Function Documentation

3.5.3.1 def hypervisor.hypervisor_factory.HypervisorFactory.get_hypervisor_instance ( ) [static]

Returns the hypervisor driver nstance.

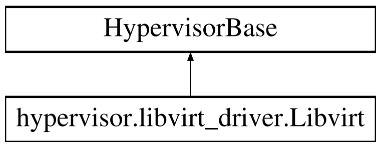
If the instance is not initialized then a RuntimeError is raised.

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/hypervisor/hypervisor_factory.py

3.6 hypervisor.libvirt_driver.Libvirt Class Reference

Inheritance diagram for hypervisor.libvirt_driver.Libvirt:



Public Member Functions

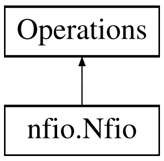
- def **deploy**
- def **pause**
- def **destroy**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/hypervisor/libvirt_driver.py

3.7 nfio.Nfio Class Reference

Inheritance diagram for nfio.Nfio:



Public Member Functions

- def __init__

    Instantiates a Nfio object.
- def **access**
- def **chmod**
- def **chown**
- def getattr

    Returns the file attributes of the file specified by path Args: path: Path of the file fh: Open file handle to the file Returns: A dictionary containing file attributes.
- def **readdir**
- def **readlink**
- def **mknod**
- def **rmdir**
- def mkdir

    The semantics have been redefined to create a new VNF instance when a directory is created under a specific type of VNF directory.
- def **statfs**
- def **unlink**
- def **symlink**
- def **rename**
- def **link**
- def **utimens**
- def **open**
- def **create**
- def read

    Reads an open file.
- def write

    Write to an open file.
- def **truncate**
- def **flush**
- def **release**
- def **fsync**

Public Attributes

- **root**
- **mountpoint**
- **hypervisor**
- **vnfs_ops**
- **module_root**

3.7.1 Constructor & Destructor Documentation

3.7.1.1 def nfio.Nfio.__init__ (*self*, *root*, *mountpoint*, *hypervisor* = 'Docker', *module_root* = 'middleboxes')

Instantiates a Nfio object.

Args: root: The root directory of nfio file system. The root directory stores persistent state about the system. mountpoint: The mountpoint of nfio file system. The mountpoint is required to intercept the file system calls via fuse. All the file system calls for fuse mounted files/directories are intercepted by libfuse and our provided implementation is executed. hypervisor: The type of hypervisor to use for deploying VNFs. The default is to use Docker containers. However, we also plan to add support for Libvirt. module_root: Root directory of the middlebox modules. Each middlebox provides it's own implementation of certain system calls in a separate module. module_root points to the root of that module. If nothing is provided a default of 'middleboxes' will be assumed. Returns: Nothing. Mounts nf.io file system at the specified mountpoint and creates a loop to act upon different file system calls.

3.7.2 Member Function Documentation

3.7.2.1 def nfio.Nfio.getattr (*self*, *path*, *fh* = None)

Returns the file attributes of the file specified by path Args: path: Path of the file fh: Open file handle to the file Returns: A dictionary containing file attributes.

The dictionary contains the following keys: st_atime: Last access time st_ctime: File creation time st_gid: Group id of the owner group st_mode: File access mode st_mtime: Last modification time st_nlink: Number of symbolic links to the file st_size: Size of the file in bytes st_uid: User id of the file owner Note: For special placeholder files for VNFs, st_size is set to a constant 1000. This is to make sure read utilities such as cat work for these special placeholder files.

3.7.2.2 def nfio.Nfio.mkdir (*self*, *path*, *mode*)

The semantics have been redefined to create a new VNF instance when a directory is created under a specific type of VNF directory.

Args: path: path of the directory to create. The path also represents the name of the new VNF instance to be created. mode: File access mode for the new directory. Returns: If path does not correspond to a directory under a specific VNF type directory then errno.EPERM is returned. Otherwise the return code is same as os.mkdir()'s return code.

3.7.2.3 def nfio.Nfio.read (*self*, *path*, *length*, *offset*, *fh*)

Reads an open file.

This nfio specific implementation parses path to see if the read is from any VNF or not. In case the read is from a VNF, the corresponding VNF module is loaded and the module's _read function is invoked to complete the read system call.

Args: path: path represents the path of the file to read from length: number of bytes to read from the file offset: byte offset indicating the starting byte to read from fh: file descriptor of the open file represented by path

Returns: length bytes from offset byte of the file represented by fh and path

Notes: VNFs can have special files which are placeholders for statistics such as number of received/sent bytes etc. VNFs provide their own implementation of read and handle reading of these special placeholder files.

3.7.2.4 def nfio.Nfio.write (*self*, *path*, *buf*, *offset*, *fh*)

Write to an open file.

In this nfio specific implementation the path is parsed to see if the write is for any specific VNF or not. If the write is for any file under a VNF directory then the corresponding VNF module is loaded and the module's _write function is invoked.

Args: path: path to the file to write buf: the data to write offset: the byte offset at which the write should begin fh: file descriptor of the open file represented by path

Returns: Returns the number of bytes written to the file starting at offset

Note: VNFs can have special files where writing specific strings trigger a specific function. For example, writing 'activate' to the 'action' file of a VNF will start the VNF. VNF specific modules handle such special cases of writing.

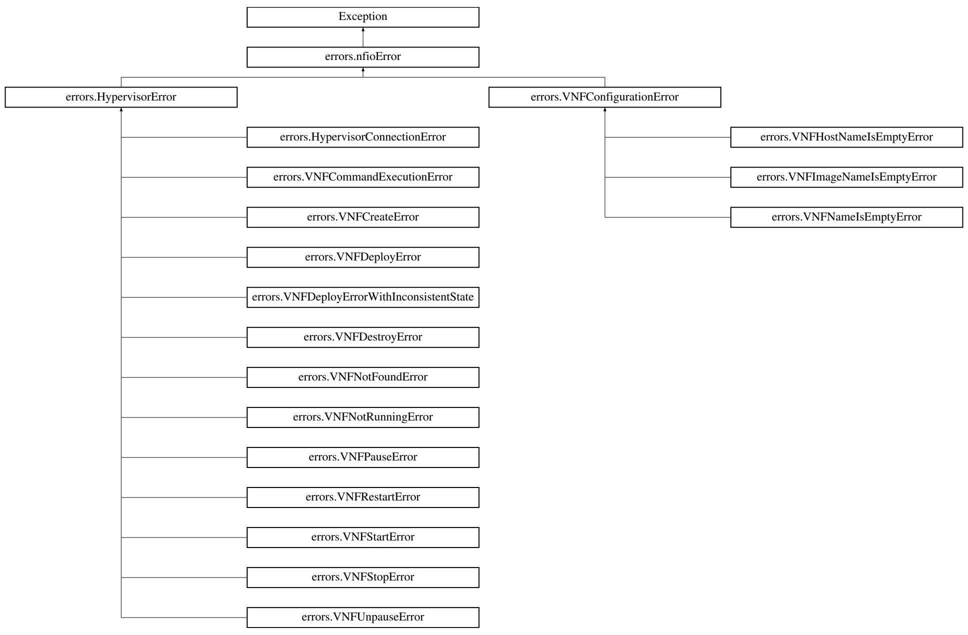
The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/nfio.py

3.8 errors.nfioError Class Reference

This module contains all the custom exceptions defined for nf.io.

Inheritance diagram for errors.nfioError:


Exception
errors.nfioError
errors.HypervisorError
errors.VNFConfigurationError
errors.HypervisorConnectionError
errors.VNFCommandExecutionError
errors.VNFCreateError
errors.VNFDeployError
errors.VNFDeployErrorWithInconsistentState
errors.VNFDestroyError
errors.VNFNotFoundError
errors.VNFNotRunningError
errors.VNFPauseError
errors.VNFRestartError
errors.VNFStartError
errors.VNFStopError
errors.VNFUnpauseError
errors.VNFHostNameIsEmptyError
errors.VNFImageNameIsEmptyError
errors.VNFNameIsEmptyError

3.8.1 Detailed Description

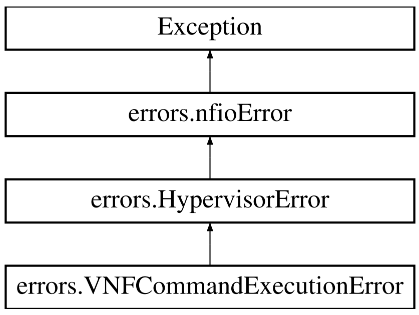
This module contains all the custom exceptions defined for nf.io.

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.9 errors.VNFCommandExecutionError Class Reference

Inheritance diagram for errors.VNFCommandExecutionError:



Public Member Functions

- def **__init__**

Public Attributes

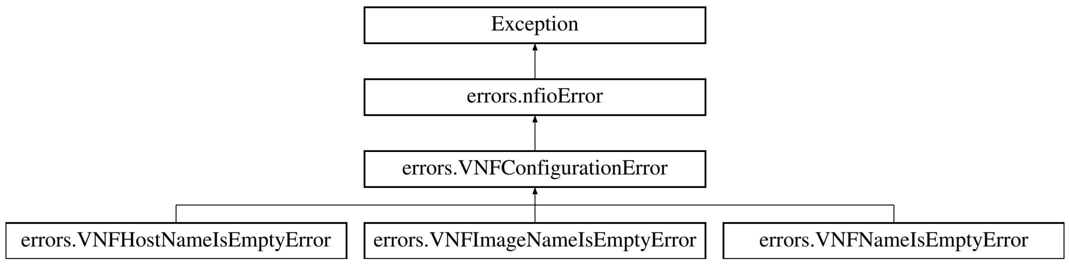
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.10 errors.VNFConfigurationError Class Reference

Inheritance diagram for errors.VNFConfigurationError:

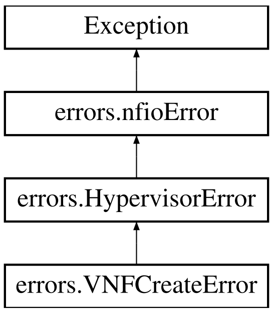

Exception
errors.nfioError
errors.VNFConfigurationError
errors.VNFHostNameIsEmptyError
errors.VNFImageNameIsEmptyError
errors.VNFNameIsEmptyError

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.11 errors.VNFCreateError Class Reference

Inheritance diagram for errors.VNFCreateError:



Public Member Functions

- def **__init__**

Public Attributes

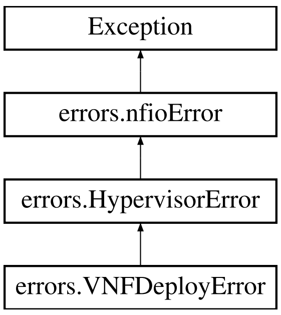
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.12 errors.VNFDeployError Class Reference

Inheritance diagram for errors.VNFDeployError:



Public Member Functions

- def **__init__**

Public Attributes

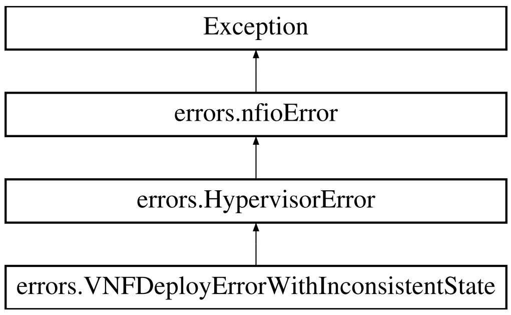
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.13 errors.VNFDeployErrorWithInconsistentState Class Reference

Inheritance diagram for errors.VNFDeployErrorWithInconsistentState:



Public Member Functions

- def **__init__**

Public Attributes

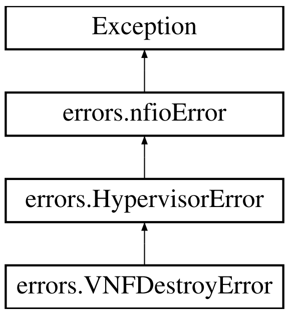
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.14 errors.VNFDestroyError Class Reference

Inheritance diagram for errors.VNFDestroyError:



Public Member Functions

- def **__init__**

Public Attributes

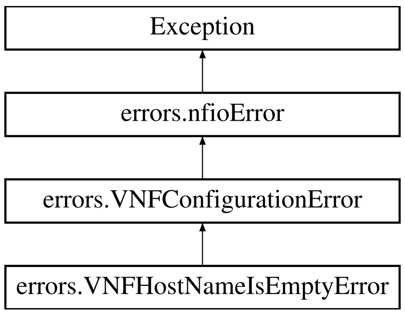
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.15 errors.VNFHostNameIsEmptyError Class Reference

Inheritance diagram for errors.VNFHostNameIsEmptyError:



Public Member Functions

- def **__init__**

Public Attributes

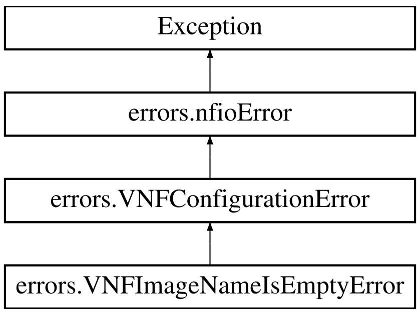
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.16 errors.VNFImageNameIsEmptyError Class Reference

Inheritance diagram for errors.VNFImageNameIsEmptyError:



Public Member Functions

- def **__init__**

Public Attributes

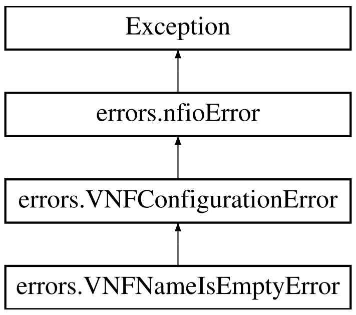
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.17 errors.VNFNameIsEmptyError Class Reference

Inheritance diagram for errors.VNFNameIsEmptyError:



Public Member Functions

- def **__init__**

Public Attributes

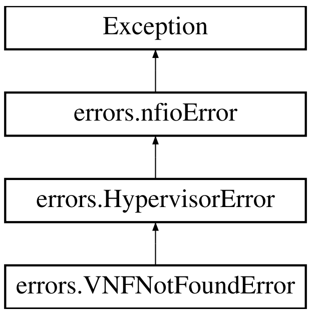
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.18 errors.VNFNotFoundError Class Reference

Inheritance diagram for errors.VNFNotFoundError:



Public Member Functions

- def **__init__**

Public Attributes

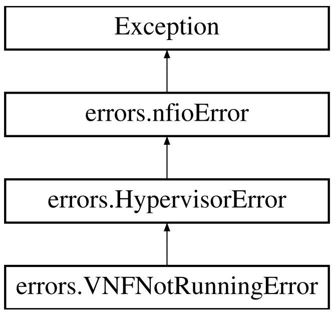
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.19 errors.VNFNotRunningError Class Reference

Inheritance diagram for errors.VNFNotRunningError:



Public Member Functions

- def **__init__**

Public Attributes

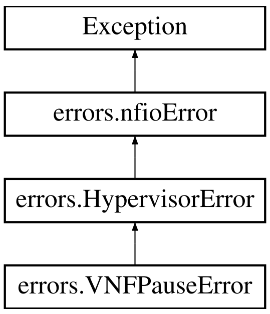
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.20 errors.VNFPauseError Class Reference

Inheritance diagram for errors.VNFPauseError:



Public Member Functions

- def **__init__**

Public Attributes

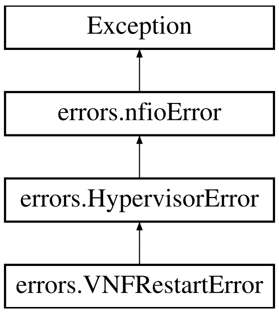
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.21 errors.VNFRestartError Class Reference

Inheritance diagram for errors.VNFRestartError:



Public Member Functions

- def **__init__**

Public Attributes

- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.22 vnfs_operations.VNFSOperations Class Reference

Provides a common set of operations for nfio.

Public Member Functions

- def **__init__**
- def vnfs_create_vnf_instance

  Create the file system structure for a VNF.
- def vnfs_get_opcode

  Determinse the type of operation based on the path.
- def vnfs_get_nf_type

  Parse the type of VNF from path.
- def vnfs_get_file_name

  Return the name of the file represented by a path.
- def vnfs_is_nf_instance

  Determines if a path represents an nf instance directory.
- def vnfs_get_instance_configuration

  Return the configuration parameters related to a VNF instance.
- def vnfs_deploy_nf

  Deploys and STARTS a VNF instance.
- def vnfs_stop_vnf

  Stops a VNF instance.

- def vnfs_start_vnf

  Starts a deployed VNF instance.
- def vnfs_destroy_vnf

  Destroys a deployed VNF instance.
- def vnfs_get_rx_bytes

  Reads the number of bytes received by a VNF instance.
- def vnfs_get_tx_bytes

  Reads the number of bytes sent by a VNF instance.
- def vnfs_get_pkt_drops

  Reads the number of packets dropped by a VNF instance.
- def vnfs_get_status

  Get the status of a VNF instance, e.g., the VNF is running/suspended/stopped etc.
- def vnfs_get_ip

  Get the status of a VNF instance, e.g., the VNF is running/suspended/stopped etc.

Public Attributes

- **vnfs_root**

Static Public Attributes

- int **OP_UNDEFINED** = 0xFF
- int **OP_NF** = 0x01

3.22.1 Detailed Description

Provides a common set of operations for nfio.

These operations act as a helper.

3.22.2 Member Function Documentation

3.22.2.1 def vnfs_operations.VNFSOperations.vnfs_create_vnf_instance (*self, path, mode*)

Create the file system structure for a VNF.

Args: path: path of the new VNF instance. mode: file creation mode for the new VNF instance directory.

Returns: returns the return code of os.mkdir

3.22.2.2 def vnfs_operations.VNFSOperations.vnfs_deploy_nf (*self, nf_path*)

Deploys and STARTS a VNF instance.

Args: nf_path: path of the VNF instance.

Returns

void

3.22.2.3 def vnfs_operations.VNFSOperations.vnfs_destroy_vnf (*self, nf_path*)

Destroys a deployed VNF instance.

Args: nf_path: path of the VNF instance.

Returns: return codes are described in hypervisor.hypervisor_return_codes module.

3.22.2.4 def vnfs_operations.VNFSOperations.vnfs_get_file_name (*self, path*)

Return the name of the file represented by a path.

Args: path: the path of the file in concern

Returns: returns the name of the file, i.e., last token after / in the path.

3.22.2.5 def vnfs_operations.VNFSOperations.vnfs_get_instance_configuration (*self, nf_path*)

Return the configuration parameters related to a VNF instance.

Args: nf_path: path of the VNF instance. e.g., /mnt/vnfsmnt/firewall/fw-alpha

Returns: A tuple representing the configuration of the VNF instance. The tuple is organized in the following order: nf_instance_name: name of the VNF instance. nf_type: type of the VNF. ip_address: IP address of the machine where this VNF will be deployed. image_name: name of the VM/container image for that VNF.

3.22.2.6 def vnfs_operations.VNFSOperations.vnfs_get_ip (*self, nf_path*)

Get the status of a VNF instance, e.g., the VNF is running/suspended/stopped etc.

Args: nf_path: path of the VNF instance.

Returns: Hypervisor specific status of the VNF. For example, if Docker is being used for VNF deployment then Docker specific container status message is returned.

3.22.2.7 def vnfs_operations.VNFSOperations.vnfs_get_nf_type (*self, path*)

Parse the type of VNF from path.

Args: path: the path of the file/directory on which some operation is being performed.

Returns: Returns the type of VNF parsed from the path, e.g., if the path is /mnt/vnfsroot/nf-types/firewall/fw-alpha/action then returns firewall.

3.22.2.8 def vnfs_operations.VNFSOperations.vnfs_get_opcode (*self, path*)

Determinse the type of operation based on the path.

Args: path: path to the file/directory on which the operation is being performed

Returns: If the file is under nf-types subdirectory in the nfio mount, then returns OP_NF. Otherwise, returns OP_U-NDEFINED.

3.22.2.9 def vnfs_operations.VNFSOperations.vnfs_get_pkt_drops (*self, nf_path*)

Reads the number of packets dropped by a VNF instance.

Args: nf_path: path of the VNF instance.

Returns: returns the number of packets dropped by a VNF instance.

3.22.2.10 def vnfs_operations.VNFSOperations.vnfs_get_rx_bytes (*self, nf_path*)

Reads the number of bytes received by a VNF instance.

Args: nf_path: path of the VNF instance.

Returns: returns the number of bytes received by a VNF instance.

3.22.2.11 def vnfs_operations.VNFSOperations.vnfs_get_status (*self, nf_path*)

Get the status of a VNF instance, e.g., the VNF is running/suspended/stopped etc.

Args: nf_path: path of the VNF instance.

Returns: Hypervisor specific status of the VNF. For example, if Docker is being used for VNF deployment then Docker specific container status message is returned.

3.22.2.12 def vnfs_operations.VNFSOperations.vnfs_get_tx_bytes (*self, nf_path*)

Reads the number of bytes sent by a VNF instance.

Args: nf_path: path of the VNF instance.

Returns: returns the number of bytes sent by a VNF instance.

3.22.2.13 def vnfs_operations.VNFSOperations.vnfs_is_nf_instance (*self, path*)

Determines if a path represents an nf instance directory.

Args: path: path of the file/directory in concern.

Returns: True: if path represents an nf instance directory. For example, if path is /mnt/vnfsmnt/nf-types/firewall/fw-alpha then returns True.

False: if the path does not represent an nf instance directory. For example, if path is /mnt/vnfsmnt/nf-types/firewall/fw-alpha/action then returns False.

3.22.2.14 def vnfs_operations.VNFSOperations.vnfs_start_vnf (*self, nf_path*)

Starts a deployed VNF instance.

Args: nf_path: path of the VNF instance.

Returns: return codes are described in hypervisor.hypervisor_return_codes module.

3.22.2.15 def vnfs_operations.VNFSOperations.vnfs_stop_vnf (*self, nf_path*)

Stops a VNF instance.

Args: nf_path: path of the VNF instance.

Returns

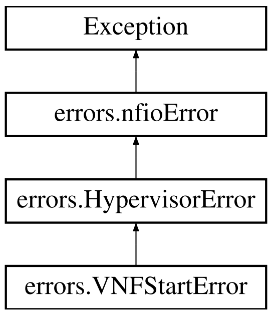
void

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/vnfs_operations.py

3.23 errors.VNFStartError Class Reference

Inheritance diagram for errors.VNFStartError:



Public Member Functions

- def **__init__**

Public Attributes

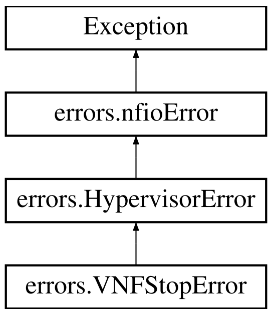
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.24 errors.VNFStopError Class Reference

Inheritance diagram for errors.VNFStopError:



Public Member Functions

- def **__init__**

Public Attributes

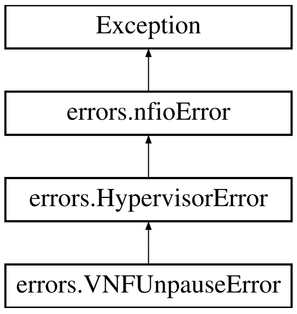
- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

3.25 errors.VNFUnpauseError Class Reference

Inheritance diagram for errors.VNFUnpauseError:



Public Member Functions

- def **__init__**

Public Attributes

- **errno**

The documentation for this class was generated from the following file:

- WatNFV/nf.io/src/errors.py

Index