AIETES Documentation

Release 0.1

Andrew Bolster

February 14, 2015

CONTENTS

Contents:

- This file is part of the Aietes Framework (https://github.com/andrewbolster/aietes)
-
- 3. Copyright 2013 Andrew Bolster (http://andrewbolster.info/) and others.
-
- All rights reserved. This program and the accompanying materials
- are made available under the terms of the Eclipse Public License v1.0
- which accompanies this distribution, and is available at
- http://www.eclipse.org/legal/epl-v10.html
-
- Contributors:
- Andrew Bolster, Queen's University Belfast (-Aug 2013), University of Liverpool (Sept 2014-)

class aietes.**Simulation**(**args*, ***kwargs*)

Defines a single simulation Keyword Arguments:

title:str(time) progress_display:bool(True) working_directory:str(/dev/shm) logtofile:str(None) logtoconsole:logging.level(INFO) logger:logging.logger(None) config_file:str(None) config:dict(None)

configure_environment(*env_config*)

Configure the physical environment within which the simulation executed Assumes empty unless told otherwise :param env_config:

configure_nodes()

Configure 'fleets' of nodes for simulation Fleets are purely logical in this case

current_state()

Returns

delta_t(*now*, *then*)

Time in seconds between two simulation times :param now: :param then:

generate_a_node(*node_name*, *node_config*)

If a node is named in the configuration file, use the defined initial vector otherwise, use configured behaviour to assign an initial vector :param node_name: :param node_config:

generate_datapackage(**args*, ***kwargs*)

Creates a bounos.DataPackage object from the current sim :param args: :param kwargs:

inner_join()

When all nodes have a move flag and none are processing

static now()

Returns

outer_join()

When all nodes have a processing flag and none are moving

classmethod populate_config(*config*, *retain_default=False*)

Parameters

- **config** –
- **retain_default** –

Returns

raise ConfigError

postprocess(*log=None*, *output_file=None*, *output_path=None*, *display_frames=None*, *data_file=False*, *movie_file=False*, *gif=False*, *input_file=None*, *plot=False*, *xres=1024*, *yres=768*, *fps=24*, *extent=True*)

Performs output and positions generation for a given simulation :param log: :param output_file: :param output_path: :param display_frames: :param data_file: :param movie_file: :param gif: :param input_file: :param plot: :param xres: :param yres: :param fps: :param extent:

prepare(*waits=False*, **args*, ***kwargs*)

Args: waits(bool): set if running interactively (i.e. sim will wait for external actions) sim_time(int): override the simulation duration

Raises: SystemExit on configuration error in setting log)

Returns: Dict: {'sim_time':prepared sim duration (int),} This is an extensible interface that can be added to but must maintain compatibility.

reverse_node_lookup(*uuid*)

Return Node Given UUID :param uuid:

simulate(*callback=None*)

Initiate the processed Simulation :param callback: Args:

callback(func): Callback function to be called at each execution step

Returns: Simulation Duration in ticks (generally seconds)

aietes.**go**(*options*, *args=None*)

Parameters

- **options** –
- **args** –

Returns

aietes.**main**()

Everyone knows what the main does; it does everything!

aietes.**option_parser**()

Returns

- This file is part of the Aietes Framework (https://github.com/andrewbolster/aietes)
-
- 3. Copyright 2013 Andrew Bolster (http://andrewbolster.info/) and others.
-
- All rights reserved. This program and the accompanying materials
- are made available under the terms of the Eclipse Public License v1.0
- which accompanies this distribution, and is available at
- http://www.eclipse.org/legal/epl-v10.html
-
- Contributors:

- Andrew Bolster, Queen's University Belfast (-Aug 2013), University of Liverpool (Sept 2014-)

class aietes.Tools.**AutoSyncShelf**(*filename*, *protocol=2*, *writeback=True*)

Parameters

- **filename** –
- **protocol** –
- **writeback** –

exception aietes.Tools.**ConfigError**(*message*, *errors=None*)

Raised when a configuration cannot be validated through ConfigObj/Validator Contains a 'status' with the boolean dict representation of the error

class aietes.Tools.**Dotdict**(*arg*, ***kwargs*)

Parameters

- **arg** –
- **kwargs** –

class aietes.Tools.**Dotdictify**(*value=None*, ***kwargs*)

Parameters

- **value** –
- **kwargs** –

Raises TypeError

class aietes.Tools.**KeepRefs**

classmethod get_instances()

class aietes.Tools.**MapEntry**(*object_id*, *position*, *velocity*, *name=None*, *distance=None*)

Parameters

- **object_id** –
- **position** –
- **velocity** –
- **name** –
- **distance** –

class aietes.Tools.**MemoryEntry**(*object_id*, *position*, *velocity*, *distance=None*, *name=None*)

Parameters

- **object_id** –
- **position** –
- **velocity** –
- **distance** –
- **name** –

class aietes.Tools.**SimTimeFilter**(*name=''*)

Brings Sim.now() into usefulness

aietes.Tools.**add_ndarray_to_set**(*ndarray*, *list*)

Parameters

- **ndarray** –
- **list** –

Returns

aietes.Tools.**agitate_position**(*position*, *maximum*, *var=10*, *minimum=None*)

Fluff a position i by randn*var, limited to maximum/minimum :param position: :param maximum: :param var: :param minimum: :return:

aietes.Tools.**angle_between**(*v1*, *v2*)

Returns the angle in radians between vectors 'v1' and 'v2':

```
>>> angle_between((1, 0, 0), (0, 1, 0))
```

1.5707963267948966 >>> angle_between((1, 0, 0), (1, 0, 0)) 0.0 >>> angle_between((1, 0, 0), (-1, 0, 0)) 3.1415926535897931

aietes.Tools.**are_equal_waypoints**(*wps*)

Compare Waypoint Objects as used by WaypointMixin ([pos],prox) Will exclude 'None' records in wps and only compare valid waypoint lists

Parameters **wps** –

aietes.Tools.**bearing**(*v*)

radian angle between a given vector and 'north' :param v:

aietes.Tools.**db2linear**(*db*)

Parameters **db** –

Returns

aietes.Tools.**distance**(*pos_a*, *pos_b*, *scale=1*)

Return the distance between two positions :param pos_a: :param pos_b: :param scale:

aietes.Tools.**fudge_normal**(*value*, *stdev*)

Parameters

- **value** –
- **stdev** –

Returns

raise ValueError

aietes.Tools.**generate_names**(*count*, *naming_convention=None*, *existing_names=None*)

Parameters

- **count** –
- **naming_convention** –
- **existing_names** –

Returns

raise ConfigError

aietes.Tools.**get_config**(*source_config=None*, *config_spec='/home/bolster/src/aietes/src/aietes/configs/default.conf'*)

Get a configuration, either using default values from aietes.configs or by taking a configobj compatible file path

Parameters

- **source_config** –
- **config_spec** –

aietes.Tools.**get_config_file**(*config*)

Return the full path to a config of a given filename if its in the default config path :param config: :return:

aietes.Tools.**get_latest_aietes_datafile**(*base_dir=None*)

Parameters **base_dir** –

Returns

raise ValueError

aietes.Tools.**grouper**(*data*)

Parameters **data** –

Returns

aietes.Tools.**is_valid_aietes_datafile**(*filename*)

Parameters **filename** –

Returns

aietes.Tools.**itersubclasses**(*cls*)

Generator over all subclasses of a given class, in depth first order.

```
>>> list(itersubclasses(int)) == [bool]
True
:param cls:
:param _seen:
>>> class A(object): pass
>>> class B(A): pass
>>> class C(A): pass
>>> class D(B,C): pass
>>> class E(D): pass
>>>
>>> for cls in itersubclasses(A):
...     print(cls.__name__)
B
D
E
C
    # get ALL (new-style) classes currently defined
    [cls.__name__ for cls in itersubclasses(object)]
['type', ...'tuple', ...]
```

aietes.Tools.**kwarger**(***kwargs*)

Parameters **kwargs** –

Returns

aietes.Tools.**linear2db**(*linear*)

Parameters **linear** –

Returns

aietes.Tools.**list_functions**(*module*)

Parameters **module** –

Returns

aietes.Tools.**listfix**(*list_type*, *value*)

Parameters

- **list_type** –
- **value** –

Returns

aietes.Tools.**literal_eval_walk**(*node*, *tabs=0*)

Parameters

- **node** –
- **tabs** –

aietes.Tools.**log_level_lookup**(*log_level*)

Parameters **log_level** –

Returns

aietes.Tools.**mag**(*vector*)
Return the magnitude of a given vector %timeit np.sqrt((uu1[0]-uu2[0])**2 +(uu1[1]-uu2[1])**2 +(uu1[2]-uu2[2])**2)-> 7.08us, %timeit np.linalg.norm(uu1-uu2) -> 11.7us. :param vector:

aietes.Tools.**memory**(*since=0.0*)
Return memory usage in Megabytes. :param since:

aietes.Tools.**mkcpickle**(*filename*, *thing*)

Parameters

- **filename** –
- **thing** –

Returns

aietes.Tools.**mkpickle**(*filename*, *thing*)

Parameters

- **filename** –
- **thing** –

Returns

aietes.Tools.**named_log**(*pos_log*)

Parameters **pos_log** –

Returns

aietes.Tools.**node_ids**(*pos_log*)

Parameters **pos_log** –

Returns

aietes.Tools.**notify_desktop**(*message*)
Thin Shim to notify when the simulations are complete, and not cry under no xsession :param message: :return:

aietes.Tools.**object_log**(*pos_log*, *object_id*)

Parameters

- **pos_log** –
- **object_id** –

Returns

aietes.Tools.**random_three_vector**()
Generates a random 3D unit vector (direction) with a uniform spherical distribution Algo from http://stackoverflow.com/questions/5408276/python-uniform-spherical-distribution :return:

aietes.Tools.**random_xy_vector**()
Generates a random 2D vector in 3D space: (Planar random walk) this is a horrible cheat but it works. :return:

aietes.Tools.**randomstr**(*length*)

Parameters **length** –

Returns

aietes.Tools.**range_grouper**(*data*)

Parameters **data** –

Returns

aietes.Tools.**records_check**(*pos_log*)

Parameters **pos_log** –

Returns

aietes.Tools.**resident**(*since=0.0*)
Return resident memory usage in Megabytes. :param since:

aietes.Tools.**results_file**(*proposed_name*, *results_dir=None*)

Parameters

- **proposed_name** –
- **results_dir** –

Returns

aietes.Tools.**sixvec**(*xyz*)

Parameters **xyz** –

Returns

aietes.Tools.**spherical_distance**(*sixvec_a*, *sixvec_b*)

Parameters

- **sixvec_a** –
- **sixvec_b** –

Returns

aietes.Tools.**stacksize**(*since=0.0*)
Return stack size in Megabytes. :param since:

aietes.Tools.**swapsize**(*since=0.0*)
Return memory usage in Megabytes. :param since:

aietes.Tools.**timeit**()

Returns

aietes.Tools.**timestamp**()

Returns

aietes.Tools.**try_forever**(*exceptions_to_catch*, *fn*)

Parameters

- **exceptions_to_catch** –
- **fn** –

Returns

aietes.Tools.**try_x_times**(*x*, *exceptions_to_catch*, *exception_to_raise*, *fn*)

Parameters

- **x** –
- **exceptions_to_catch** –
- **exception_to_raise** –
- **fn** –

Returns

raise exception_to_raise

aietes.Tools.**uncpickle**(*filename*)

Parameters **filename** –

Returns

aietes.Tools.**unext**(*filename*)

Parameters **filename** –

Returns

aietes.Tools.**unit**(*vector*)
Return the unit vector :param vector:

aietes.Tools.**unpickle**(*filename*)

Parameters **filename** –

Returns

aietes.Tools.**update_dict**(*d*, *keys*, *value*, *safe=False*)

Parameters

- **d** –
- **keys** –
- **value** –

- **safe** –

Raises KeyError

aietes.Tools.**validate_config**(*config=None*, *final_check=False*)

Generate valid confobj configuration information by interpolating a given config file with the defaults

Parameters

- **config** –
- **final_check** –

NOTE: This does not verify if any of the functionality requested in the config is THERE Only that the config 'makes sense' as requested.

I.e. does not check if particular modular behaviour exists or not.

- This file is part of the Aietes Framework (https://github.com/andrewbolster/aietes)
-
- 3. Copyright 2013 Andrew Bolster (http://andrewbolster.info/) and others.
-
- All rights reserved. This program and the accompanying materials
- are made available under the terms of the Eclipse Public License v1.0
- which accompanies this distribution, and is available at
- http://www.eclipse.org/legal/epl-v10.html
-
- Contributors:
- Andrew Bolster, Queen's University Belfast (-Aug 2013), University of Liverpool (Sept 2014-)

class bounos.**BounosModel**(**args*, ***kwargs*)

BounosModel acts as an interactive superclass of DataPackage, designed for interactive simulation/analysis

It is blankly initialised with no arguments and must be initialised by either interacting with a simulation (update_data_from_sim) or from an existing datafile (import_datafile)

update_data_from_sim(*p*, *v*, *names*, *environment*, *now*)

Call back function used by SimulationStep if doing real time simulation

Imports DataPackage data from the running simulation up to the requested time (self.t)

bounos.**add_achievements**(*ax*, *d*, *annotate_achievements=False*)

Parameters

- **ax** –
- **d** –
- **annotate_achievements** –

bounos.**custom_fusion_run**(*args*, *data*, *title*)

Parameters

- **args** –
- **data** –

- **title** –

bounos.**custom_metric_run**(*args*, *data*, *title*)

Parameters

- **args** –
- **data** –
- **title** –

bounos.**custom_parser**()

Returns

bounos.**detect_and_identify**(*d*)

Parameters **d** –

Returns

bounos.**generate_sources**(*sources*, *comms_only=False*)

From a given list of DataPackage-able sources, yield a title/content tuple based on their d.title

Parameters

- **sources** –
- **comms_only** – Only return the comms substructure rather than the full datapackage

Return data

bounos.**global_adjust**(*figure*, *axes*, *scale=2*)

Parameters

- **figure** –
- **axes** –
- **scale** –

General Figure adjustments: Subplot-spacing adjustments Figure sizing/scaling

Args: figure(Figure): figure to be adjusted scale(int/float): adjust figure to scale (optional:2)

bounos.**load_sources**(*sources*, *comms_only=False*)

From a given list of DataPackage-able sources, parallelize their instantiation as a names dict based on their d.title

Parameters

- **sources** –
- **comms_only** – Only return the comms substructure rather than the full datapackage

Return data

bounos.**main**()

Initial Entry Point; Does very little other that option parsing Raises:

ValueError if graph selection doesn't make any sense

bounos.**multirun**(*args*, *basedir='.'*)

Parameters

- **basedir** –

- **args** –

Returns

bounos.**npz_in_dir**(*path*)

From a given dir, return all the NPZs :param path: :return sources:

bounos.**plot_detections**(*ax*, *metric*, *orig_data*, *shade_region=False*, *real_culprits=None*, *good_behaviour='Waypoint'*)

Plot Detection Overlay including False-positive analysis.

Will attempt heuristic analysis of 'real' culprit from DataPackage behaviour records

Parameters

- **ax** –
- **metric** –
- **orig_data** –
- **shade_region** –
- **real_culprits** –
- **good_behaviour** –

Args: ax(axes): plot to operate on metric(Metric): metric to use for detection orig_data(DataPackage): data used shade_region(bool): shade the detection region (optional:False) real_culprits(list): provide a list of culprits for false-positive testing (optional) good_behaviour(str): override the default good behaviour (optional: "Waypoint")

bounos.**print_analysis**(*d*)

Parameters **d** –

Returns

bounos.**run_detection_fusion**(*data*, *args=None*)

Generate a trust fusion across available metrics, and plot both the metric deviations, per-metric detections, and the trust fusion per node, per dataset

Parameters

- **data** –
- **args** –

Args: data(list of DataPackage): datasets to plot horizontally args(argparse.NameSpace): formatting and option arguments (optional)

bounos.**run_metric_comparison**(*data*, *args=None*)

Generate available metrics, and plot both the metric values, per-metric detections, per dataset

Parameters

- **data** –
- **args** –

Args: data(list of DataPackage): datasets to plot horizontally args(argparse.NameSpace): formatting and option arguments (optional)

bounos.**run_overlay**(*data*, *args=None*)

Parameters

- **data** –
- **args** –

Args: data(list of DataPackage): datasets to plot horizontally args(argparse.NameSpace): formatting and option arguments (optional)

- This file is part of the Aietes Framework (https://github.com/andrewbolster/aietes)
-
- 3. Copyright 2013 Andrew Bolster (http://andrewbolster.info/) and others.
-
- All rights reserved. This program and the accompanying materials
- are made available under the terms of the Eclipse Public License v1.0
- which accompanies this distribution, and is available at
- http://www.eclipse.org/legal/epl-v10.html
-
- Contributors:
- Andrew Bolster, Queen's University Belfast (-Aug 2013), University of Liverpool (Sept 2014-)
- This file is part of the Aietes Framework
- (https://github.com/andrewbolster/aietes)
-
- 3. Copyright 2013 Andrew Bolster (http://andrewbolster.info/) and others.
-
- All rights reserved. This program and the accompanying materials
- are made available under the terms of the Eclipse Public License v1.0
- which accompanies this distribution, and is available at
- http://www.eclipse.org/legal/epl-v10.html
-
- Contributors:
- Andrew Bolster, Queen's University Belfast (-Aug 2013), University of Liverpool (Sept 2014-)

class polybos.**ExperimentManager**(*node_count=None*, *title=None*, *parallel=False*, *base_config_file=None*, *base_exp_path=None*, **args*, ***kwargs*)

Parameters

- **node_count** –
- **title** –
- **parallel** –
- **base_config_file** –

- **base_exp_path** –
- **args** –
- **kwargs** –

add_application_variable_scenario(*variable*, *value_range*, *title_range=None*)
Add a scenario with a range of application/Node configuration values to the experimental run

This *UPDATES* the default nodes rather than adding custom ones

Parameters

- **variable** –
- **value_range** –
- **title_range** –

Args: variable(str): mutable value description value_range(range or generator): values to be tested against.

add_default_scenario(*runcount=1*, *title=None*)
Stick to the defaults :param runcount: :param title:

add_minority_n_behaviour_suite(*behaviour_list*, *n_minority=1*, *title='Behaviour'*)
Generate scenarios based on a list of 'attacking' behaviours, i.e. minority behaviours

Parameters

- **behaviour_list** –
- **n_minority** –
- **title** –

Args: behaviour_list(list): minority behaviours n_minority(int): number of minority attackers in each scenario (optional)

add_position_scaling_range(*scale_range*, *title=None*, *basis_node_name='n1'*, *scale_environment=True*, *base_scenario=None*)
Using the base_config_file, generate a range of scaled positions for nodes that are manually set (i.e. operates only on the 'initial_position' value

ONLY DEALS IN 2D AND ASSUMES ALL Z-VALUES ARE THE SAME :param scale_range: :param basis_node_name: :param title: :return:

add_ratio_scenarios(*badbehaviour*, *goodbehaviour=None*)
Add scenarios based on a ratio of behaviours of identical nodes

If goodbehaviour is not specified, then the default node configuration *should* be used for the remaining nodes

Parameters

- **badbehaviour** –
- **goodbehaviour** –

Args: badbehaviour(str):Aietes behaviour definition string (i.e. modulename) goodbehaviour(str):Aietes behaviour definition string (i.e. modulename) (optional)

add_variable_2_range_scenarios(*v_dict*)
Add a 2dim range of scenarios based on a dictionary of value ranges. This generates a meshgrid and samples scenarios across the 2dim space

Parameters **v_dict** –

Args: v_dict(dict):{'variable':'value_range', 'variable':'value_range'}

add_variable_node_scenario(*node_range*)
Add a scenario with a range of configuration values to the experimental run

Parameters **node_range** –

Args: variable(str): mutable value description value_range(range or generator): values to be tested against.

add_variable_range_scenario(*variable*, *value_range*, *title_range=None*)
Add a scenario with a range of configuration values to the experimental run

Parameters

- **variable** –
- **value_range** –
- **title_range** –

Args: variable(str): mutable value description value_range(range or generator): values to be tested against.

dump_analysis()
Ignore actual simulation information, record trust analysis stats to a pickle

dump_dataruns()
Dump scenarios into the exp_path directory

dump_self()
Attempt to dump the entire experiment state

generate_simulation_stats()

Returns: List of scenario stats (i.e. list of lists of run statistics dicts)

static print_stats(*experiment*, *verbose=False*)

Perform and print a range of summary experiment statistics including Fleet Distance (sum of velocities), Fleet Efficiency (Distance per time per node), Stdev(INDA) (Proxy for fleet positional variability) Stdev(INDD) (Proxy for fleet positional efficiency) Max Achievement Count, Percentage completion rate (how much of the fleet got the top count)

Parameters

- **experiment** –
- **verbose** –

run(*runtime=None*, *runcount=None*, *retain_data=True*, *queue=False*, ***kwargs*)
Construct an execution environment and farm off simulation to scenarios :param runtime: :param runcount: :param retain_data: :param kwargs: Args:

runtime(int): Override simulation duration (normally inherited from config) runcount(int): Number of repeated executions of this scenario; this value overrides the

value set on init

retain_data(bool/str): Decides wether the scenario state data is maintained or allowed to be GC'd
can be one of [True,'file','additional_only']

update_all_nodes(*config_dict*)
Applys a behaviour (given as a string) to the experimental default for node generation :param config_dict: Args:

behaviour(str): new default behaviour

update_default_behaviour(*behaviour*)
Applys a behaviour (given as a string) to the experimental default for node generation :param behaviour: Args:

behaviour(str): new default behaviour

update_default_node(*config_dict*)
Applys a behaviour (given as a string) to the experimental default for node generation :param config_dict: Args:

behaviour(str): new default behaviour

update_duration(*tmax*)
Update the simulation time of currently configured scenarios :param tmax: Args:

tmax(int): update experiment simulation duration for all scenarios

update_environment(*environment*)
Update the environment extent of currently configured scenarios :param environment: Args:

environment([int,int,int]): update experiment simulation environment for all scenarios

update_node_counts(*new_count*)
Updates the node-count across all scenarios

Parameters **new_count** –

Args: new_count(int):new value to be used across scenarios

class polybos.**PseudoScenario**
PseudoScenario(title, datarun)

datarun
Alias for field number 1

title
Alias for field number 0

class polybos.**RecoveredExperiment**(*dirpath*)
SubClass to recover a partially executed experiment from an experiment directory.

Not guaranteed to work in any way what so ever. :param dirpath: :return:

classmethod walk_dir(*path*)

Parameters **path** –

Returns

class polybos.**Scenario**(*default_config=None*, *default_config_file=None*, *runcount=1*, *title=None*, **args*, ***kwargs*)

Scenario Object

The Scenario Object deals with config management and passthrough, as well as some optional execution characteristics. The purpose of this manager is to abstract as much as humanly possible.

add_custom_node(*variable_map*, *count=1*)

Adds a node to the scenario based on a dict of mutable key,values :param variable_map: :param count: Args:

variable_map(dict): variables and values to be modified from the default count(int): if set, creates count instances of the custom node

add_default_node(*count=1*)

Adds a default node :param count: Args:

count(int): if set, creates count instances of the default node

add_node(*node_conf*, *names=None*, *count=1*)

Adds a node to the scenario based on a (hopefully valid) node configuration :param node_conf: :param names: :param count: Args:

node_conf(dict): Fully defined node config dict names(list(str)): List of names for new nodes count(int): if set, creates count instances of the node

Raises: RuntimeError if name definition doesn't make sense

commit()

'Lock' the scenario, generating the final config, filling in any 'empty' config sections Raises:

RuntimeError: on attempting to commit and already committed scenario

generate_config()

Generate a config dict from the current state of the planned scenario Returns:

DataPackage compatible dict

generate_configobj()

Generate a ConfigObj from the current state of the planned scenario Returns:

DataPackage compatible ConfigObj

generate_run_stats(*sim_run_dataset=None*)

Recieving a bounos.datapackage, generate relevant stats This is nasty and I can't remember why I did it this way :param sim_run_dataset: Returns:

A list of dict's given from DataPackage.package_statistics()

get_behaviour_dict()

Generate and return a dict of currently configured behaviours wrt names of nodes eg. {'Waypoing':['alpha','beta','gamma'],'Flock':['omega']}

Returns: dict of behaviours associated with a list of node names

run(*runcount=None*, *runtime=None*, **args*, ***kwargs*)

Offload this to AIETES :param runcount: :param runtime: :param args: :param kwargs: Args:

run_parallel(*runcount=None*, *runtime=None*, *queueing_pool=False*, ***kwargs*)

Offload this to AIETES multiprocessing queue

Parameters

- **runtime** –

- **kwargs** –
- **runcount** –

Args:

runcount(int): Number of repeated executions of this scenario; this value overrides the value set on init

runtime(int): Override simulation duration (normally inherited from config)

set_duration(*tmax*)
Set the scenario simulation duration, :param tmax: Args:

tmax(int): New simulation time

set_environment(*environment*)
Set the scenario simulation environment extent, :param environment: Args:

environment([int,int,int]): New simulation environment extent

set_node_count(*count*)

Set the scenario node count, but does not update the configuration (this is satisfied in the commit method)

Parameters count –

Args: count(int): New Node count

update_default_node(*variable*, *value*)
Update the default node for the scenario. :param variable: :param value: Args:

variable(str):The Variable to be modified (should me in the mutable map) value: the value to set that variable to

Raises: RuntimeError if attempting to modify after commit.

update_node(*node_conf*, *mutable*, *value*)

Used to update selected field mappings between scenario definition and the scenario configspec, as defined in mutable_node_configs

Parameters

- **node_conf** –
- **mutable** –
- **value** –

Args: node_conf(dict): current node configuration to be updated mutable(str): a string describing the aspect to be changed, present in the mutable map value(any): the mutable value to be set

Raises: NotImplementedError: on invalid mutable key

write()
Dump the datafiles into a path but creating a folder with our name

CHAPTER

ONE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

a

b

e

p

V

W