

Cheetah/Savanna API

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	codar Namespace Reference	9
5.2	codar.cheetah Namespace Reference	9
5.2.1	Detailed Description	9
5.3	codar.cheetah.adios2_interface Namespace Reference	10
5.3.1	Detailed Description	10
5.3.2	Function Documentation	10
5.3.2.1	get_adios_version()	10
5.3.2.2	set_engine()	10
5.3.2.3	set_transport()	11
5.3.2.4	set_var_operation()	11
5.4	codar.cheetah.adios_params Namespace Reference	11
5.4.1	Detailed Description	11

5.4.2	Function Documentation	12
5.4.2.1	adios_xml_transform()	12
5.4.2.2	adios_xml_transport()	12
5.4.2.3	xml_has_transport()	12
5.5	codar.cheetah.config Namespace Reference	13
5.5.1	Detailed Description	13
5.5.2	Function Documentation	13
5.5.2.1	etc_path()	13
5.5.2.2	get_dataspaces_num_servers()	13
5.5.2.3	machine_submit_env_path()	14
5.5.2.4	scheduler_path()	14
5.5.3	Variable Documentation	14
5.5.3.1	CHEETAH_PATH_MACHINE_CONFIG	14
5.5.3.2	CHEETAH_PATH_SCHEDULER	14
5.5.3.3	CODAR_PATH	14
5.5.3.4	DATA_PATH	15
5.5.3.5	PACKAGE_PATH	15
5.5.3.6	WORKFLOW_SCRIPT	15
5.6	codar.cheetah.exc Namespace Reference	15
5.6.1	Detailed Description	15
5.7	codar.cheetah.helpers Namespace Reference	15
5.7.1	Function Documentation	16
5.7.1.1	copy_to_dir()	16
5.7.1.2	copy_to_path()	16
5.7.1.3	copytree_to_dir()	16
5.7.1.4	dir_size()	17
5.7.1.5	get_file_size()	17
5.7.1.6	get_immediate_subdirs()	17
5.7.1.7	is_campaign_directory()	17
5.7.1.8	is_executable()	18

5.7.1.9	json_config_set_option()	18
5.7.1.10	make_executable()	18
5.7.1.11	parse_timedelta_seconds()	18
5.7.1.12	relative_or_absolute_path()	19
5.7.1.13	relative_or_absolute_path_list()	19
5.7.1.14	require_campaign_directory()	19
5.7.1.15	swift_escape_string()	19
5.8	codar.cheetah.launchers Namespace Reference	19
5.8.1	Detailed Description	20
5.8.2	Variable Documentation	20
5.8.2.1	TAU_PROFILE_PATTERN	20
5.9	codar.cheetah.loader Namespace Reference	20
5.9.1	Detailed Description	20
5.9.2	Function Documentation	20
5.9.2.1	load_experiment_class()	20
5.10	codar.cheetah.machine_launchers Namespace Reference	21
5.10.1	Function Documentation	21
5.10.1.1	get_launcher()	21
5.10.2	Variable Documentation	21
5.10.2.1	machine_launchers	21
5.11	codar.cheetah.model Namespace Reference	21
5.11.1	Detailed Description	22
5.11.2	Variable Documentation	22
5.11.2.1	RESERVED_CODE_NAMES	22
5.12	codar.cheetah.parameters Namespace Reference	22
5.12.1	Detailed Description	22
5.13	codar.cheetah.pbs Namespace Reference	23
5.13.1	Detailed Description	23
5.13.2	Function Documentation	23
5.13.2.1	open_pbs_file()	23

5.13.2.2	write_run_script()	24
5.13.3	Variable Documentation	24
5.13.3.1	PBS_FORMAT_TEMPLATE	24
5.13.3.2	PBS_NAME	24
5.13.3.3	SUBMIT_FORMAT_TEMPLATE	24
5.14	codar.cheetah.report_generator Namespace Reference	25
5.14.1	Detailed Description	25
5.14.2	Function Documentation	25
5.14.2.1	generate_report()	25
5.15	codar.cheetah.runners Namespace Reference	25
5.15.1	Detailed Description	26
5.16	codar.cheetah.status Namespace Reference	26
5.16.1	Detailed Description	26
5.16.2	Function Documentation	26
5.16.2.1	get_workflow_status()	26
5.16.2.2	print_campaign_status()	26
5.17	codar.cheetah.templates Namespace Reference	27
5.17.1	Detailed Description	27
5.17.2	Variable Documentation	27
5.17.2.1	CAMPAIGN_ENV_TEMPLATE	27
5.17.2.2	GROUP_ENV_TEMPLATE	27
5.18	codar.savanna Namespace Reference	28
5.18.1	Detailed Description	28
5.19	codar.savanna.consumer Namespace Reference	28
5.19.1	Detailed Description	28
5.20	codar.savanna.exc Namespace Reference	28
5.20.1	Detailed Description	29
5.21	codar.savanna.machines Namespace Reference	29
5.21.1	Detailed Description	29
5.21.2	Function Documentation	29

5.21.2.1	get_by_name()	29
5.21.3	Variable Documentation	29
5.21.3.1	cori	30
5.21.3.2	local	30
5.21.3.3	SCHEDULER_OPTIONS	30
5.21.3.4	summit	30
5.21.3.5	theta	31
5.21.3.6	titan	31
5.22	codar.savanna.main Namespace Reference	31
5.22.1	Detailed Description	31
5.22.2	Function Documentation	31
5.22.2.1	get_job_id()	32
5.22.2.2	main()	32
5.22.2.3	parse_args()	32
5.22.3	Variable Documentation	32
5.22.3.1	consumer	32
5.23	codar.savanna.model Namespace Reference	32
5.23.1	Detailed Description	33
5.23.2	Variable Documentation	33
5.23.2.1	KILL_WAIT	33
5.23.2.2	RETURN_NAME	33
5.23.2.3	STDERR_NAME	33
5.23.2.4	STDOUT_NAME	33
5.23.2.5	WAIT_DELAY_GIVE_UP	34
5.23.2.6	WAIT_DELAY_KILL	34
5.23.2.7	WALLTIME_NAME	34
5.24	codar.savanna.node_layout Namespace Reference	34
5.25	codar.savanna.producer Namespace Reference	34
5.25.1	Detailed Description	34
5.26	codar.savanna.runners Namespace Reference	34

5.26.1	Variable Documentation	35
5.26.1.1	aprun	35
5.26.1.2	jsrun	35
5.26.1.3	mpiexec	35
5.26.1.4	srun	35
5.27	codar.savanna.scheduler Namespace Reference	35
5.27.1	Detailed Description	36
5.28	codar.savanna.status Namespace Reference	36
5.28.1	Detailed Description	36
5.28.2	Variable Documentation	36
5.28.2.1	DONE	36
5.28.2.2	KILLED	37
5.28.2.3	NOT_STARTED	37
5.28.2.4	REASON_EXCEPTION	37
5.28.2.5	REASON_FAILED	37
5.28.2.6	REASON_NOFIT	37
5.28.2.7	REASON_SUCCEEDED	37
5.28.2.8	REASON_TIMEOUT	38
5.28.2.9	RUNNING	38
5.29	codar.savanna.summit_helper Namespace Reference	38
5.29.1	Function Documentation	38
5.29.1.1	create_erf_file()	38
5.29.1.2	get_nodes_reqd()	38

6	Class Documentation	39
6.1	codar.cheetah.report_generator._ReportGenerator Class Reference	39
6.1.1	Detailed Description	39
6.1.2	Constructor & Destructor Documentation	39
6.1.2.1	__init__()	40
6.1.3	Member Function Documentation	40
6.1.3.1	parse_campaign()	40
6.1.3.2	parse_run_dir()	40
6.1.3.3	parse_sweep_group()	40
6.1.3.4	parse_user_campaigns()	41
6.1.3.5	write_output()	41
6.1.4	Member Data Documentation	41
6.1.4.1	campaign_directory	41
6.1.4.2	current_campaign_user	41
6.1.4.3	output_filename	41
6.1.4.4	parsed_runs	42
6.1.4.5	run_status	42
6.1.4.6	unique_keys	42
6.1.4.7	user_run_script	42
6.2	codar.cheetah.report_generator._RunParser Class Reference	42
6.2.1	Detailed Description	43
6.2.2	Constructor & Destructor Documentation	43
6.2.2.1	__init__()	43
6.2.3	Member Function Documentation	43
6.2.3.1	execute_user_run_script()	43
6.2.3.2	get_cheetah_perf_data()	44
6.2.3.3	get_rc_names()	44
6.2.3.4	get_run_params()	44
6.2.3.5	read_adios_output_file_sizes()	44
6.2.3.6	read_fob_json()	44

6.2.3.7	read_node_layout()	45
6.2.3.8	read_sos_perf_data()	45
6.2.3.9	serialize_params_nested_dict()	45
6.2.3.10	verify_run_successful()	45
6.2.4	Member Data Documentation	46
6.2.4.1	exit_status	46
6.2.4.2	fob_dict	46
6.2.4.3	rc_name_exe	46
6.2.4.4	rc_names	46
6.2.4.5	rc_working_dir	46
6.2.4.6	run_dir	47
6.2.4.7	serialized_run_params	47
6.2.4.8	user_run_script	47
6.3	codar.cheetah.model.Campaign Class Reference	47
6.3.1	Detailed Description	49
6.3.2	Constructor & Destructor Documentation	49
6.3.2.1	__init__()	49
6.3.3	Member Function Documentation	49
6.3.3.1	make_experiment_run_dir()	49
6.3.4	Member Data Documentation	49
6.3.4.1	app_config_scripts	50
6.3.4.2	app_dir	50
6.3.4.3	codes [1/2]	50
6.3.4.4	codes [2/2]	50
6.3.4.5	inputs [1/2]	50
6.3.4.6	inputs [2/2]	50
6.3.4.7	kill_on_partial_failure	51
6.3.4.8	machine	51
6.3.4.9	machine_app_config_script	51
6.3.4.10	machine_scheduler_options	51

6.3.4.11	name	51
6.3.4.12	post_process_script	51
6.3.4.13	python_path	52
6.3.4.14	run_dir_setup_script	52
6.3.4.15	run_post_process_script	52
6.3.4.16	run_post_process_stop_group_on_failure	52
6.3.4.17	runs	52
6.3.4.18	scheduler_options	52
6.3.4.19	sos_analysis_path	53
6.3.4.20	sosd_num_aggregators	53
6.3.4.21	sosd_path	53
6.3.4.22	supported_machines	53
6.3.4.23	sweeps	53
6.3.4.24	tau_config	53
6.3.4.25	umask	54
6.4	codar.cheetah.exc.CampaignParseError Class Reference	54
6.4.1	Detailed Description	55
6.5	codar.cheetah.exc.CheetahException Class Reference	55
6.5.1	Detailed Description	56
6.6	codar.cheetah.parameters.CodeCommand Class Reference	56
6.6.1	Detailed Description	57
6.6.2	Constructor & Destructor Documentation	57
6.6.2.1	__init__()	57
6.6.3	Member Function Documentation	57
6.6.3.1	add_arg()	57
6.6.3.2	add_option()	58
6.6.3.3	get_argv()	58
6.6.4	Member Data Documentation	58
6.6.4.1	args	58
6.6.4.2	options	58

6.6.4.3	target	58
6.7	codar.cheetah.parameters.Instance Class Reference	59
6.7.1	Detailed Description	60
6.7.2	Constructor & Destructor Documentation	60
6.7.2.1	__init__()	60
6.7.3	Member Function Documentation	60
6.7.3.1	add_parameter()	60
6.7.3.2	as_dict()	60
6.7.3.3	as_string()	61
6.7.3.4	code_commands()	61
6.7.3.5	get_codes_argv()	61
6.7.3.6	get_hostfile()	61
6.7.3.7	get_nprocs()	62
6.7.3.8	get_parameter_values_by_type()	62
6.7.3.9	get_sched_opts()	62
6.7.3.10	parameter_values()	62
6.8	codar.savanna.scheduler.JobList Class Reference	63
6.8.1	Detailed Description	63
6.8.2	Constructor & Destructor Documentation	64
6.8.2.1	__init__()	64
6.8.3	Member Function Documentation	64
6.8.3.1	__len__()	64
6.8.3.2	add_job()	64
6.8.3.3	pop_job()	64
6.9	codar.savanna.producer.JSONFilePipelineReader Class Reference	65
6.9.1	Detailed Description	65
6.9.2	Constructor & Destructor Documentation	66
6.9.2.1	__init__()	66
6.9.3	Member Function Documentation	66
6.9.3.1	read_pipelines()	66

6.9.4	Member Data Documentation	66
6.9.4.1	file_path	66
6.10	codar.cheetah.launchers.Launcher Class Reference	67
6.10.1	Detailed Description	68
6.10.2	Constructor & Destructor Documentation	68
6.10.2.1	__init__()	68
6.10.3	Member Function Documentation	68
6.10.3.1	create_group_directory()	69
6.10.3.2	read_jobid()	69
6.10.4	Member Data Documentation	69
6.10.4.1	batch_script_name	69
6.10.4.2	batch_walltime_name	70
6.10.4.3	jobid_file_name	70
6.10.4.4	machine_name	70
6.10.4.5	name	70
6.10.4.6	num_codes	70
6.10.4.7	output_directory	70
6.10.4.8	run_command_name	71
6.10.4.9	run_json_name	71
6.10.4.10	run_out_name	71
6.10.4.11	runner_name	71
6.10.4.12	scheduler_name	71
6.10.4.13	status_script_name	71
6.10.4.14	submit_out_name	72
6.10.4.15	submit_script_name	72
6.10.4.16	wait_script_name	72
6.11	codar.savanna.machines.Machine Class Reference	72
6.11.1	Detailed Description	73
6.11.2	Constructor & Destructor Documentation	73
6.11.2.1	__init__()	74

6.11.3	Member Function Documentation	74
6.11.3.1	get_nodes_reqd()	74
6.11.3.2	get_scheduler_options()	74
6.11.4	Member Data Documentation	74
6.11.4.1	dataspaces_servers_per_node	74
6.11.4.2	name	75
6.11.4.3	node_class	75
6.11.4.4	node_exclusive	75
6.11.4.5	processes_per_node	75
6.11.4.6	runner_name	75
6.11.4.7	scheduler_name	75
6.11.4.8	scheduler_options	76
6.12	codar.savanna.machines.MachineNode Class Reference	76
6.12.1	Detailed Description	76
6.12.2	Constructor & Destructor Documentation	76
6.12.2.1	__init__()	77
6.12.3	Member Function Documentation	77
6.12.3.1	to_json()	77
6.12.3.2	validate_layout()	77
6.12.4	Member Data Documentation	77
6.12.4.1	cpu	77
6.12.4.2	gpu	77
6.13	codar.savanna.exc.MachineNotFound Class Reference	78
6.13.1	Detailed Description	78
6.13.2	Constructor & Destructor Documentation	79
6.13.2.1	__init__()	79
6.14	codar.cheetah.exc.MachineNotFound Class Reference	79
6.14.1	Detailed Description	80
6.14.2	Constructor & Destructor Documentation	80
6.14.2.1	__init__()	80

6.15	codar.savanna.runners.MPIRunner Class Reference	81
6.15.1	Detailed Description	82
6.15.2	Constructor & Destructor Documentation	82
6.15.2.1	__init__()	82
6.15.3	Member Function Documentation	82
6.15.3.1	wrap()	82
6.15.4	Member Data Documentation	82
6.15.4.1	exe	82
6.15.4.2	hostfile	83
6.15.4.3	nodes_arg	83
6.15.4.4	nprocs_arg	83
6.15.4.5	tasks_per_node_arg	83
6.16	codar.savanna.model.NodeConfig Class Reference	83
6.16.1	Detailed Description	84
6.16.2	Constructor & Destructor Documentation	84
6.16.2.1	__init__()	84
6.16.3	Member Data Documentation	84
6.16.3.1	cpu	84
6.16.3.2	gpu	84
6.16.3.3	num_ranks_per_node	84
6.17	codar.savanna.node_layout.NodeLayout Class Reference	85
6.17.1	Detailed Description	86
6.17.2	Constructor & Destructor Documentation	86
6.17.2.1	__init__()	86
6.17.3	Member Function Documentation	86
6.17.3.1	add_node()	86
6.17.3.2	as_data_list()	87
6.17.3.3	codes_per_node()	87
6.17.3.4	copy()	87
6.17.3.5	default_no_share_layout()	87

6.17.3.6	get_node_containing_code()	87
6.17.3.7	group_codes_by_node()	88
6.17.3.8	populate_remaining()	88
6.17.3.9	ppn()	88
6.17.3.10	serialize_to_dict()	88
6.17.3.11	shared_nodes()	89
6.17.3.12	validate()	89
6.17.4	Member Data Documentation	89
6.17.4.1	layout_list	89
6.17.4.2	layout_map	89
6.18	codar.cheetah.parameters.Param Class Reference	90
6.18.1	Detailed Description	91
6.18.2	Constructor & Destructor Documentation	91
6.18.2.1	__init__()	91
6.18.3	Member Function Documentation	91
6.18.3.1	__get__()	92
6.18.3.2	__len__()	92
6.18.4	Member Data Documentation	92
6.18.4.1	name	92
6.18.4.2	target	92
6.18.4.3	values	92
6.19	codar.cheetah.parameters.ParamADIOS2XML Class Reference	93
6.19.1	Detailed Description	94
6.19.2	Constructor & Destructor Documentation	94
6.19.2.1	__init__()	94
6.19.3	Member Data Documentation	94
6.19.3.1	io_name	95
6.19.3.2	operation_name	95
6.19.3.3	rc	95
6.19.3.4	values	95

6.20	codar.cheetah.parameters.ParamAdiosXML Class Reference	96
6.20.1	Detailed Description	97
6.20.2	Constructor & Destructor Documentation	97
6.20.2.1	__init__()	97
6.20.3	Member Data Documentation	97
6.20.3.1	group_name	97
6.20.3.2	param_type	98
6.20.3.3	var_name	98
6.21	codar.cheetah.parameters.ParamCmdLineArg Class Reference	98
6.21.1	Detailed Description	99
6.21.2	Constructor & Destructor Documentation	99
6.21.2.1	__init__()	99
6.21.3	Member Data Documentation	100
6.21.3.1	position	100
6.22	codar.cheetah.parameters.ParamCmdLineOption Class Reference	100
6.22.1	Detailed Description	101
6.22.2	Constructor & Destructor Documentation	101
6.22.2.1	__init__()	101
6.22.3	Member Data Documentation	102
6.22.3.1	option	102
6.23	codar.cheetah.parameters.ParamConfig Class Reference	102
6.23.1	Detailed Description	103
6.23.2	Constructor & Destructor Documentation	103
6.23.2.1	__init__()	104
6.23.3	Member Data Documentation	104
6.23.3.1	config_filename	104
6.23.3.2	match_string	104
6.24	codar.cheetah.parameters.ParamEnvVar Class Reference	104
6.24.1	Detailed Description	105
6.24.2	Constructor & Destructor Documentation	105

6.24.2.1	<code>__init__()</code>	105
6.24.3	Member Data Documentation	106
6.24.3.1	<code>option</code>	106
6.25	<code>codar.cheetah.parameters.ParameterValue</code> Class Reference	106
6.25.1	Detailed Description	107
6.25.2	Constructor & Destructor Documentation	107
6.25.2.1	<code>__init__()</code>	107
6.25.3	Member Function Documentation	107
6.25.3.1	<code>__getattr__()</code>	107
6.25.3.2	<code>is_type()</code>	108
6.25.4	Member Data Documentation	108
6.25.4.1	<code>value</code>	108
6.26	<code>codar.cheetah.parameters.ParamKeyValue</code> Class Reference	108
6.26.1	Detailed Description	109
6.26.2	Constructor & Destructor Documentation	109
6.26.2.1	<code>__init__()</code>	110
6.26.3	Member Data Documentation	110
6.26.3.1	<code>config_filename</code>	110
6.26.3.2	<code>key_name</code>	110
6.27	<code>codar.cheetah.parameters.ParamRunner</code> Class Reference	110
6.27.1	Detailed Description	111
6.27.2	Constructor & Destructor Documentation	111
6.27.2.1	<code>__init__()</code>	111
6.28	<code>codar.cheetah.parameters.ParamSchedulerArgs</code> Class Reference	112
6.28.1	Detailed Description	113
6.28.2	Constructor & Destructor Documentation	113
6.28.2.1	<code>__init__()</code>	113
6.29	<code>codar.savanna.model.Pipeline</code> Class Reference	113
6.29.1	Detailed Description	115
6.29.2	Constructor & Destructor Documentation	115

6.29.2.1	<code>__init__()</code>	115
6.29.3	Member Function Documentation	115
6.29.3.1	<code>add_done_callback()</code>	115
6.29.3.2	<code>add_fatal_callback()</code>	115
6.29.3.3	<code>force_kill_all()</code>	116
6.29.3.4	<code>from_data()</code>	116
6.29.3.5	<code>get_nodes_used()</code>	116
6.29.3.6	<code>get_pids()</code>	116
6.29.3.7	<code>get_state()</code>	117
6.29.3.8	<code>join_all()</code>	117
6.29.3.9	<code>remove_done_callback()</code>	117
6.29.3.10	<code>remove_fatal_callback()</code>	117
6.29.3.11	<code>run_finished()</code>	117
6.29.3.12	<code>run_post_process_script()</code>	118
6.29.3.13	<code>set_ppn()</code>	118
6.29.3.14	<code>set_total_nodes()</code>	118
6.29.3.15	<code>start()</code>	118
6.29.4	Member Data Documentation	118
6.29.4.1	<code>done_callbacks</code>	119
6.29.4.2	<code>fatal_callbacks</code>	119
6.29.4.3	<code>id</code>	119
6.29.4.4	<code>kill_on_partial_failure</code>	119
6.29.4.5	<code>launch_mode</code>	119
6.29.4.6	<code>log_prefix</code>	119
6.29.4.7	<code>machine_name</code>	120
6.29.4.8	<code>node_layout</code>	120
6.29.4.9	<code>nodes_assigned</code>	120
6.29.4.10	<code>post_process_args</code>	120
6.29.4.11	<code>post_process_script</code>	120
6.29.4.12	<code>post_process_stop_on_failure</code>	120

6.29.4.13 runs	121
6.29.4.14 total_nodes	121
6.29.4.15 total_procs	121
6.29.4.16 working_dir	121
6.30 codar.savanna.consumer.PipelineRunner Class Reference	122
6.30.1 Detailed Description	123
6.30.2 Constructor & Destructor Documentation	123
6.30.2.1 __init__()	123
6.30.3 Member Function Documentation	123
6.30.3.1 add_pipeline()	124
6.30.3.2 kill_all()	124
6.30.3.3 pipeline_fatal()	124
6.30.3.4 pipeline_finished()	124
6.30.3.5 run_finished()	125
6.30.3.6 run_pipelines()	125
6.30.3.7 stop()	125
6.30.4 Member Data Documentation	125
6.30.4.1 allocated_nodes	125
6.30.4.2 free_cv	126
6.30.4.3 free_nodes	126
6.30.4.4 job_list	126
6.30.4.5 job_list_cv	126
6.30.4.6 machine_name	126
6.30.4.7 max_nodes	126
6.30.4.8 pipelines	127
6.30.4.9 pipelines_lock	127
6.30.4.10 ppn	127
6.30.4.11 runner	127
6.31 codar.savanna.status.PipelineState Class Reference	128
6.31.1 Detailed Description	128

6.31.2	Constructor & Destructor Documentation	129
6.31.2.1	__init__()	129
6.31.3	Member Function Documentation	129
6.31.3.1	as_data()	129
6.31.4	Member Data Documentation	129
6.31.4.1	id	129
6.31.4.2	reason	129
6.31.4.3	return_codes	130
6.31.4.4	state	130
6.32	codar.savanna.model.Run Class Reference	130
6.32.1	Detailed Description	132
6.32.2	Constructor & Destructor Documentation	132
6.32.2.1	__init__()	132
6.32.3	Member Function Documentation	132
6.32.3.1	add_callback()	132
6.32.3.2	close()	133
6.32.3.3	create_node_config()	133
6.32.3.4	exception()	133
6.32.3.5	from_data()	133
6.32.3.6	get_nodes_used()	134
6.32.3.7	get_pid()	134
6.32.3.8	get_returncode()	134
6.32.3.9	join()	134
6.32.3.10	kill()	134
6.32.3.11	killed()	135
6.32.3.12	mpmd_run()	135
6.32.3.13	remove_callback()	135
6.32.3.14	run()	135
6.32.3.15	set_runner()	135
6.32.3.16	succeeded()	136

6.32.3.17	<code>timed_out()</code>	136
6.32.4	Member Data Documentation	136
6.32.4.1	<code>args</code>	136
6.32.4.2	<code>callbacks</code>	136
6.32.4.3	<code>depends_on_runs</code>	136
6.32.4.4	<code>env</code>	137
6.32.4.5	<code>erf_file</code>	137
6.32.4.6	<code>exe</code>	137
6.32.4.7	<code>hostfile</code>	137
6.32.4.8	<code>log_prefix</code>	137
6.32.4.9	<code>machine</code>	137
6.32.4.10	<code>name</code>	138
6.32.4.11	<code>node_config</code>	138
6.32.4.12	<code>nodes</code>	138
6.32.4.13	<code>nodes_assigned</code>	138
6.32.4.14	<code>nprocs</code>	138
6.32.4.15	<code>res_set</code>	138
6.32.4.16	<code>return_path</code>	139
6.32.4.17	<code>runner</code>	139
6.32.4.18	<code>runner_override</code>	139
6.32.4.19	<code>sched_args</code>	139
6.32.4.20	<code>sleep_after</code>	139
6.32.4.21	<code>stderr_path</code>	139
6.32.4.22	<code>stdout_path</code>	140
6.32.4.23	<code>tasks_per_node</code>	140
6.32.4.24	<code>timeout</code>	140
6.32.4.25	<code>walltime_path</code>	140
6.32.4.26	<code>working_dir</code>	140
6.33	<code>codar.cheetah.model.Run</code> Class Reference	141
6.33.1	Detailed Description	142

6.33.2	Constructor & Destructor Documentation	142
6.33.2.1	__init__()	142
6.33.3	Member Function Documentation	142
6.33.3.1	add_dataspaces_support()	143
6.33.3.2	get_app_param_dict()	143
6.33.3.3	get_fob_data_list()	143
6.33.3.4	get_total_nprocs()	143
6.33.3.5	insert_sosflow()	144
6.33.4	Member Data Documentation	144
6.33.4.1	codes	144
6.33.4.2	codes_path	144
6.33.4.3	component_inputs	144
6.33.4.4	component_subdirs	144
6.33.4.5	inputs	145
6.33.4.6	instance	145
6.33.4.7	machine	145
6.33.4.8	node_layout	145
6.33.4.9	run_components	145
6.33.4.10	run_id	145
6.33.4.11	run_path	146
6.33.4.12	sosflow_analysis	146
6.33.4.13	sosflow_profiling	146
6.33.4.14	total_nodes	146
6.34	codar.cheetah.model.RunComponent Class Reference	147
6.34.1	Detailed Description	148
6.34.2	Constructor & Destructor Documentation	148
6.34.2.1	__init__()	148
6.34.3	Member Function Documentation	148
6.34.3.1	as_fob_data()	148
6.34.4	Member Data Documentation	149

6.34.4.1	adios_xml_file	149
6.34.4.2	after_rc_done	149
6.34.4.3	args	149
6.34.4.4	component_inputs	149
6.34.4.5	env	149
6.34.4.6	exe	150
6.34.4.7	hostfile	150
6.34.4.8	linked_with_sosflow	150
6.34.4.9	name	150
6.34.4.10	nprocs	150
6.34.4.11	runner_override	150
6.34.4.12	sched_args	151
6.34.4.13	sleep_after	151
6.34.4.14	timeout	151
6.34.4.15	working_dir	151
6.35	codar.savanna.runners.Runner Class Reference	152
6.35.1	Detailed Description	152
6.35.2	Member Function Documentation	152
6.35.2.1	wrap()	153
6.36	codar.cheetah.runners.Runner Class Reference	153
6.36.1	Detailed Description	154
6.36.2	Member Function Documentation	154
6.36.2.1	wrap_app_command()	154
6.36.3	Member Data Documentation	154
6.36.3.1	name	154
6.37	codar.cheetah.runners.RunnerCray Class Reference	155
6.37.1	Detailed Description	156
6.37.2	Member Function Documentation	156
6.37.2.1	wrap_app_command()	156
6.37.3	Member Data Documentation	156

6.37.3.1	name	156
6.38	codar.cheetah.runners.RunnerLocal Class Reference	157
6.38.1	Detailed Description	158
6.38.2	Member Function Documentation	158
6.38.2.1	wrap_app_command()	158
6.38.3	Member Data Documentation	158
6.38.3.1	name	158
6.39	codar.savanna.exc.SavannaException Class Reference	159
6.39.1	Detailed Description	159
6.40	codar.savanna.machines.SummitNode Class Reference	160
6.40.1	Detailed Description	160
6.40.2	Constructor & Destructor Documentation	160
6.40.2.1	__init__()	161
6.40.3	Member Function Documentation	161
6.40.3.1	to_json()	161
6.40.3.2	validate_layout()	161
6.41	codar.cheetah.parameters.SummitOpts Class Reference	161
6.41.1	Detailed Description	161
6.41.2	Constructor & Destructor Documentation	162
6.41.2.1	__init__()	162
6.42	codar.savanna.runners.SummitRunner Class Reference	162
6.42.1	Detailed Description	163
6.42.2	Constructor & Destructor Documentation	163
6.42.2.1	__init__()	164
6.42.3	Member Function Documentation	164
6.42.3.1	wrap()	164
6.42.3.2	wrap_deprecated()	164
6.42.4	Member Data Documentation	164
6.42.4.1	bind_arg	164
6.42.4.2	cpus_per_rs_arg	165

6.42.4.3	exe	165
6.42.4.4	gpus_per_rs_arg	165
6.42.4.5	launch_distribution_arg	165
6.42.4.6	machine	165
6.42.4.7	nrs_arg	165
6.42.4.8	rs_per_host_arg	166
6.42.4.9	tasks_per_rs_arg	166
6.43	codar.cheetah.parameters.Sweep Class Reference	166
6.43.1	Detailed Description	167
6.43.2	Constructor & Destructor Documentation	167
6.43.2.1	__init__()	167
6.43.3	Member Function Documentation	167
6.43.3.1	get_instances()	167
6.43.4	Member Data Documentation	168
6.43.4.1	node_layout	168
6.43.4.2	parameters	168
6.43.4.3	rc_dependency	168
6.44	codar.cheetah.parameters.SweepGroup Class Reference	168
6.44.1	Detailed Description	169
6.44.2	Constructor & Destructor Documentation	170
6.44.2.1	__init__()	170
6.44.3	Member Data Documentation	170
6.44.3.1	component_inputs	170
6.44.3.2	component_subdirs	170
6.44.3.3	launch_mode	170
6.44.3.4	max_procs	171
6.44.3.5	name	171
6.44.3.6	nodes	171
6.44.3.7	parameter_groups	171
6.44.3.8	per_run_timeout	171

6.44.3.9	run_repetitions	171
6.44.3.10	sosflow_analysis	172
6.44.3.11	sosflow_profiling	172
6.44.3.12	walltime	172
6.45	codar.cheetah.parameters.SymLink Class Reference	172
6.45.1	Detailed Description	173
6.45.2	Constructor & Destructor Documentation	173
6.45.2.1	__init__()	173
6.45.3	Member Data Documentation	173
6.45.3.1	source	174
6.46	codar.savanna.status.WorkflowStatus Class Reference	174
6.46.1	Detailed Description	175
6.46.2	Constructor & Destructor Documentation	175
6.46.2.1	__init__()	175
6.46.3	Member Function Documentation	175
6.46.3.1	set_state()	175
6.46.4	Member Data Documentation	175
6.46.4.1	file_path	175
7	File Documentation	177
7.1	__init__.py File Reference	177
7.2	cheetah/__init__.py File Reference	177
7.3	savanna/__init__.py File Reference	177
7.4	cheetah/adios2_interface.py File Reference	177
7.5	cheetah/adios_params.py File Reference	178
7.6	cheetah/config.py File Reference	178
7.7	cheetah/exc.py File Reference	178
7.8	savanna/exc.py File Reference	179
7.9	cheetah/helpers.py File Reference	179
7.10	cheetah/launchers.py File Reference	179
7.11	cheetah/loader.py File Reference	180

7.12	cheetah/machine_launchers.py File Reference	180
7.13	cheetah/model.py File Reference	180
7.14	savanna/model.py File Reference	181
7.15	cheetah/parameters.py File Reference	181
7.16	cheetah/pbs.py File Reference	182
7.17	cheetah/report_generator.py File Reference	182
7.18	cheetah/runners.py File Reference	182
7.19	savanna/runners.py File Reference	183
7.20	cheetah/status.py File Reference	183
7.21	savanna/status.py File Reference	183
7.22	cheetah/templates.py File Reference	184
7.23	savanna/consumer.py File Reference	184
7.24	savanna/machines.py File Reference	184
7.25	savanna/main.py File Reference	185
7.26	savanna/node_layout.py File Reference	185
7.27	savanna/producer.py File Reference	186
7.28	savanna/scheduler.py File Reference	186
7.29	savanna/summit_helper.py File Reference	186
Index		187

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

codar	9
codar.cheetah	9
codar.cheetah.adios2_interface	10
codar.cheetah.adios_params	11
codar.cheetah.config	13
codar.cheetah.exc	15
codar.cheetah.helpers	15
codar.cheetah.launchers	19
codar.cheetah.loader	20
codar.cheetah.machine_launchers	21
codar.cheetah.model	21
codar.cheetah.parameters	22
codar.cheetah.pbs	23
codar.cheetah.report_generator	25
codar.cheetah.runners	25
codar.cheetah.status	26
codar.cheetah.templates	27
codar.savanna	28
codar.savanna.consumer	28
codar.savanna.exc	28
codar.savanna.machines	29
codar.savanna.main	31
codar.savanna.model	32
codar.savanna.node_layout	34
codar.savanna.producer	34
codar.savanna.runners	34
codar.savanna.scheduler	35
codar.savanna.status	36
codar.savanna.summit_helper	38

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

codar.cheetah.report_generator._ReportGenerator	39
codar.cheetah.report_generator._RunParser	42
Exception	
codar.cheetah.exc.CheetahException	55
codar.cheetah.exc.CampaignParseError	54
codar.cheetah.exc.MachineNotFound	79
codar.savanna.exc.SavannaException	159
codar.savanna.exc.MachineNotFound	78
codar.savanna.machines.MachineNode	76
codar.savanna.machines.SummitNode	160
codar.savanna.model.NodeConfig	83
object	
codar.cheetah.launchers.Launcher	67
codar.cheetah.model.Campaign	47
codar.cheetah.model.Run	141
codar.cheetah.model.RunComponent	147
codar.cheetah.parameters.CodeCommand	56
codar.cheetah.parameters.Instance	59
codar.cheetah.parameters.Param	90
codar.cheetah.parameters.ParamADIOS2XML	93
codar.cheetah.parameters.ParamAdiosXML	96
codar.cheetah.parameters.ParamCmdLineArg	98
codar.cheetah.parameters.ParamCmdLineOption	100
codar.cheetah.parameters.ParamConfig	102
codar.cheetah.parameters.ParamEnvVar	104
codar.cheetah.parameters.ParamKeyValue	108
codar.cheetah.parameters.ParamRunner	110
codar.cheetah.parameters.ParamSchedulerArgs	112
codar.cheetah.parameters.ParameterValue	106
codar.cheetah.parameters.Sweep	166
codar.cheetah.parameters.SweepGroup	168
codar.cheetah.runners.Runner	153
codar.cheetah.runners.RunnerCray	155
codar.cheetah.runners.RunnerLocal	157

codar.savanna.consumer.PipelineRunner	122
codar.savanna.machines.Machine	72
codar.savanna.model.Pipeline	113
codar.savanna.node_layout.NodeLayout	85
codar.savanna.producer.JSONFilePipelineReader	65
codar.savanna.runners.Runner	152
codar.savanna.runners.MPIRunner	81
codar.savanna.runners.SummitRunner	162
codar.savanna.scheduler.JobList	63
codar.savanna.status.PipelineState	128
str	
codar.cheetah.parameters.SymLink	172
codar.cheetah.parameters.SummitOpts	161
Thread	
codar.savanna.model.Run	130
codar.savanna.status.WorkflowStatus	174

Chapter 3

Class Index

3.1 Class List

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

codar.cheetah.report_generator._ReportGenerator	39
codar.cheetah.report_generator._RunParser	42
codar.cheetah.model.Campaign	47
codar.cheetah.exc.CampaignParseError	54
codar.cheetah.exc.CheetahException	55
codar.cheetah.parameters.CodeCommand	56
codar.cheetah.parameters.Instance	59
codar.savanna.scheduler.JobList	63
codar.savanna.producer.JSONFilePipelineReader	65
codar.cheetah.launchers.Launcher	67
codar.savanna.machines.Machine	72
codar.savanna.machines.MachineNode	76
codar.savanna.exc.MachineNotFound	78
codar.cheetah.exc.MachineNotFound	79
codar.savanna.runners.MPIRunner	81
codar.savanna.model.NodeConfig	83
codar.savanna.node_layout.NodeLayout	85
codar.cheetah.parameters.Param	90
codar.cheetah.parameters.ParamADIOS2XML	93
codar.cheetah.parameters.ParamAdiosXML	96
codar.cheetah.parameters.ParamCmdLineArg	98
codar.cheetah.parameters.ParamCmdLineOption	100
codar.cheetah.parameters.ParamConfig	102
codar.cheetah.parameters.ParamEnvVar	104
codar.cheetah.parameters.ParameterValue	106
codar.cheetah.parameters.ParamKeyValue	108
codar.cheetah.parameters.ParamRunner	110
codar.cheetah.parameters.ParamSchedulerArgs	112
codar.savanna.model.Pipeline	113
codar.savanna.consumer.PipelineRunner	122
codar.savanna.status.PipelineState	128
codar.savanna.model.Run	130
codar.cheetah.model.Run	141
codar.cheetah.model.RunComponent	147
codar.savanna.runners.Runner	152

codar.cheetah.runners.Runner	153
codar.cheetah.runners.RunnerCray	155
codar.cheetah.runners.RunnerLocal	157
codar.savanna.exc.SavannaException	159
codar.savanna.machines.SummitNode	160
codar.cheetah.parameters.SummitOpts	161
codar.savanna.runners.SummitRunner	162
codar.cheetah.parameters.Sweep	166
codar.cheetah.parameters.SweepGroup	168
codar.cheetah.parameters.SymLink	172
codar.savanna.status.WorkflowStatus	174

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

__init__.py	177
cheetah/__init__.py	177
cheetah/adios2_interface.py	177
cheetah/adios_params.py	178
cheetah/config.py	178
cheetah/exc.py	178
cheetah/helpers.py	179
cheetah/launchers.py	179
cheetah/loader.py	180
cheetah/machine_launchers.py	180
cheetah/model.py	180
cheetah/parameters.py	181
cheetah/pbs.py	182
cheetah/report_generator.py	182
cheetah/runners.py	182
cheetah/status.py	183
cheetah/templates.py	184
savanna/__init__.py	177
savanna/consumer.py	184
savanna/exc.py	179
savanna/machines.py	184
savanna/main.py	185
savanna/model.py	181
savanna/node_layout.py	185
savanna/producer.py	186
savanna/runners.py	183
savanna/scheduler.py	186
savanna/status.py	183
savanna/submit_helper.py	186

Chapter 5

Namespace Documentation

5.1 codar Namespace Reference

Namespaces

- [cheetah](#)
- [savanna](#)

5.2 codar.cheetah Namespace Reference

Namespaces

- [adios2_interface](#)
- [adios_params](#)
- [config](#)
- [exc](#)
- [helpers](#)
- [launchers](#)
- [loader](#)
- [machine_launchers](#)
- [model](#)
- [parameters](#)
- [pbs](#)
- [report_generator](#)
- [runners](#)
- [status](#)
- [templates](#)

5.2.1 Detailed Description

Import most important classes into top level cheetah module namespace.

5.3 codar.cheetah.adios2_interface Namespace Reference

Functions

- def [get_adios_version](#) (xml_file)
- def [set_engine](#) (xmlfile, io_obj, engine_type, parameters=None)
- def [set_transport](#) (xmlfile, io_obj, transport_type, parameters=None)
- def [set_var_operation](#) (xmlfile, io_obj, var_name, operation, parameters=None)

5.3.1 Detailed Description

ADIOS2 Interface

5.3.2 Function Documentation

5.3.2.1 [get_adios_version\(\)](#)

```
def codar.cheetah.adios2_interface.get_adios_version (
    xml_file )
```

Get the ADIOS version of this xml file.

:param xml_file: Path to the adios xml file
:return: 1 (adios version 1) or 2 (adios version 2)

Definition at line 32 of file adios2_interface.py.

5.3.2.2 [set_engine\(\)](#)

```
def codar.cheetah.adios2_interface.set_engine (
    xmlfile,
    io_obj,
    engine_type,
    parameters = None )
```

Set the engine type for an input IO object.

:param xmlfile: String. The ADIOS2 xml file to be modified
:param io_obj: String. Name of the io object which contains the engine
:param engine_type: String. The engine type to be set for the io object
:param parameters: List. A list of dicts containing 'key' and 'value' keys
:return: True on success, False on error

Definition at line 51 of file adios2_interface.py.

5.3.2.3 set_transport()

```
def codar.cheetah.adios2_interface.set_transport (
    xmlfile,
    io_obj,
    transport_type,
    parameters = None )
```

Set the transport type for an io object

:param xmlfile: String. The ADIOS2 xml file to be modified
:param io_obj: String. Name of the io object that contains the engine
:param transport_type String. The transport type for this io object
:param parameters: A dict containing the parameter keys and values
:return: True on success, False on error

Definition at line 76 of file adios2_interface.py.

5.3.2.4 set_var_operation()

```
def codar.cheetah.adios2_interface.set_var_operation (
    xmlfile,
    io_obj,
    var_name,
    operation,
    parameters = None )
```

Set an operation on a variable

:param xmlfile: String. The ADIOS2 xml file to be modified
:param io_obj: String. Name of the io object that contains the engine
:param var_name String. Name of the variable
:param operation String. The operation to be performed on the variable
:param parameters: A dict containing the parameter keys and values
:return: True on success, False on error

Definition at line 100 of file adios2_interface.py.

5.4 codar.cheetah.adios_params Namespace Reference

Functions

- def [adios_xml_transform](#) (xml_filepath, group_name, var_name, value)
- def [adios_xml_transport](#) (xml_filepath, group_name, method_name, method_opts)
- def [xml_has_transport](#) (xml_filepath, transport_type)

5.4.1 Detailed Description

Functions for parsing and editing the ADIOS xml file to enable variable transforms. Transforms include compression and reduction. 'Transform' is an ADIOS specific term.

5.4.2 Function Documentation

5.4.2.1 `adios_xml_transform()`

```
def codar.cheetah.adios_params.adios_xml_transform (
    xml_filepath,
    group_name,
    var_name,
    value )
```

Edit the ADIOS XML file to enable transform (compression/reduction) for a variable

```
:param group_name:  Name of the variable that will be transformed
:param var_name:    Name of the variable that will be transformed
:param value:       Transform type and options (sz, zfp etc.)
:param xml_filepath: Absolute path of the adios xml file. This will be in
                    the run directory.
```

TODO: add error handling, tests, e.g. for when variable not found in XML file

Definition at line 10 of file `adios_params.py`.

5.4.2.2 `adios_xml_transport()`

```
def codar.cheetah.adios_params.adios_xml_transport (
    xml_filepath,
    group_name,
    method_name,
    method_opts )
```

Definition at line 33 of file `adios_params.py`.

5.4.2.3 `xml_has_transport()`

```
def codar.cheetah.adios_params.xml_has_transport (
    xml_filepath,
    transport_type )
```

Check ADIOS XML file if provided transport method is present for any group

```
:param xml_filepath: Path to the adios xml file
:param transport_type: Transport type (POSIX, MPI, MPI_AGGREGATE,
DATASPACE, DIMES, FLEXPATH)
:return: True if transport type found, else false.
```

Definition at line 42 of file `adios_params.py`.

5.5 codar.cheetah.config Namespace Reference

Functions

- def [scheduler_path](#) (scheduler_name)
- def [machine_submit_env_path](#) (machine_name)
- def [etc_path](#) (conf_name)
- def [get_dataspaces_num_servers](#) (num_dimes_clients, num_dataspaces_clients)

Variables

- [PACKAGE_PATH](#) = os.path.realpath(os.path.dirname(__file__))
- [DATA_PATH](#) = os.path.join([PACKAGE_PATH](#), "data")
- [CODAR_PATH](#) = os.path.realpath(os.path.join([PACKAGE_PATH](#), ".."))
- [CHEETAH_PATH_SCHEDULER](#) = os.path.join([DATA_PATH](#), "scheduler")
- [CHEETAH_PATH_MACHINE_CONFIG](#) = os.path.join([DATA_PATH](#), "machine_config")
- [WORKFLOW_SCRIPT](#) = os.path.join([CODAR_PATH](#), "savanna", "main.py")

5.5.1 Detailed Description

Cheetah paths and (in future) features for loading site configuration.

5.5.2 Function Documentation

5.5.2.1 [etc_path\(\)](#)

```
def codar.cheetah.config.etc_path (  
    conf_name )
```

Definition at line 29 of file config.py.

5.5.2.2 [get_dataspaces_num_servers\(\)](#)

```
def codar.cheetah.config.get_dataspaces_num_servers (  
    num_dimes_clients,  
    num_dataspaces_clients )
```

Get the number of dataspaces server instances that must be created for a given number of client processes.

Definition at line 33 of file config.py.

5.5.2.3 machine_submit_env_path()

```
def codar.cheetah.config.machine_submit_env_path (
    machine_name )
```

Definition at line 24 of file config.py.

5.5.2.4 scheduler_path()

```
def codar.cheetah.config.scheduler_path (
    scheduler_name )
```

Definition at line 20 of file config.py.

5.5.3 Variable Documentation

5.5.3.1 CHEETAH_PATH_MACHINE_CONFIG

```
codar.cheetah.config.CHEETAH_PATH_MACHINE_CONFIG = os.path.join(DATA\_PATH, "machine_config")
```

Definition at line 15 of file config.py.

5.5.3.2 CHEETAH_PATH_SCHEDULER

```
codar.cheetah.config.CHEETAH_PATH_SCHEDULER = os.path.join(DATA\_PATH, "scheduler")
```

Definition at line 13 of file config.py.

5.5.3.3 CODAR_PATH

```
codar.cheetah.config.CODAR_PATH = os.path.realpath(os.path.join(PACKAGE\_PATH, ".."))
```

Definition at line 11 of file config.py.

5.5.3.4 DATA_PATH

```
codar.cheetah.config.DATA_PATH = os.path.join(PACKAGE_PATH, "data")
```

Definition at line 9 of file config.py.

5.5.3.5 PACKAGE_PATH

```
codar.cheetah.config.PACKAGE_PATH = os.path.realpath(os.path.dirname(__file__))
```

Definition at line 8 of file config.py.

5.5.3.6 WORKFLOW_SCRIPT

```
codar.cheetah.config.WORKFLOW_SCRIPT = os.path.join(CODAR_PATH, "savanna", "main.py")
```

Definition at line 17 of file config.py.

5.6 codar.cheetah.exc Namespace Reference

Classes

- class [CampaignParseError](#)
- class [CheetahException](#)
- class [MachineNotFound](#)

5.6.1 Detailed Description

Exceptions.

5.7 codar.cheetah.helpers Namespace Reference

Functions

- def [make_executable](#) (path)
- def [swift_escape_string](#) (s)
- def [parse_timedelta_seconds](#) (v)
- def [copy_to_dir](#) (source_file, dest_dir, follow_symlinks=True)
- def [copy_to_path](#) (source_file, dest_file, follow_symlinks=True)
- def [is_executable](#) (fpath)
- def [copytree_to_dir](#) (source_dir, dest_dir, follow_symlinks=True)
- def [relative_or_absolute_path](#) (prefix, path)
- def [relative_or_absolute_path_list](#) (prefix, path_list)
- def [get_immediate_subdirs](#) (dir_path)
- def [dir_size](#) (path)
- def [get_file_size](#) (dir_entry)
- def [is_campaign_directory](#) (path)
- def [require_campaign_directory](#) (path)
- def [json_config_set_option](#) (filename, key, value)

5.7.1 Function Documentation

5.7.1.1 `copy_to_dir()`

```
def codar.cheetah.helpers.copy_to_dir (
    source_file,
    dest_dir,
    follow_symlinks = True )
```

Wrapper around `copyfile` with directory destination and more control over permissions.

Definition at line 80 of file `helpers.py`.

5.7.1.2 `copy_to_path()`

```
def codar.cheetah.helpers.copy_to_path (
    source_file,
    dest_file,
    follow_symlinks = True )
```

Wrapper around `copyfile` that respects `umask` and preserves executability.

Definition at line 94 of file `helpers.py`.

5.7.1.3 `copytree_to_dir()`

```
def codar.cheetah.helpers.copytree_to_dir (
    source_dir,
    dest_dir,
    follow_symlinks = True )
```

Custom version of `copytree` that does not preserve permissions, but does preserve executability. The goal is to respect the current `umask` but keep executable files executable.

Definition at line 112 of file `helpers.py`.

5.7.1.4 dir_size()

```
def codar.cheetah.helpers.dir_size (
    path )
```

Get the size of the directory represented by path recursively.
:param path: Path to the dir whose size needs to be calculated
:return: size in bytes of the dir

Definition at line 151 of file helpers.py.

5.7.1.5 get_file_size()

```
def codar.cheetah.helpers.get_file_size (
    dir_entry )
```

Get size of the file or directory pointed to by path.
Directory size is recursive; it includes sizes of enclosing files/dirs.
:param dir_entry: path to the file or directory. Should not contain wildcards.
 Must be of type DirEntry.
:return: size in bytes

Definition at line 170 of file helpers.py.

5.7.1.6 get_immediate_subdirs()

```
def codar.cheetah.helpers.get_immediate_subdirs (
    dir_path )
```

Get a list of top-level subdirectories.
:param dir_path: Directory path to search
:return: list of subdirectory names

Definition at line 141 of file helpers.py.

5.7.1.7 is_campaign_directory()

```
def codar.cheetah.helpers.is_campaign_directory (
    path )
```

Return True if the specified path exists, is a directory, and has a .campaign file to indicate it's a top level campaign directory.

Definition at line 186 of file helpers.py.

5.7.1.8 is_executable()

```
def codar.cheetah.helpers.is_executable (
    fpath )
```

Definition at line 107 of file helpers.py.

5.7.1.9 json_config_set_option()

```
def codar.cheetah.helpers.json_config_set_option (
    filename,
    key,
    value )
```

Definition at line 201 of file helpers.py.

5.7.1.10 make_executable()

```
def codar.cheetah.helpers.make_executable (
    path )
```

Definition at line 14 of file helpers.py.

5.7.1.11 parse_timedelta_seconds()

```
def codar.cheetah.helpers.parse_timedelta_seconds (
    v )
```

Parse a time duration. Can be a number of seconds (integer only), a timedelta object, or a string in "HH:MM:SS" format. Returns the number of seconds in the duration as an int, safe for JSON serialization or passing to time.sleep.

```
>>> parse_timedelta_seconds('15')
15
>>> parse_timedelta_seconds('01:15')
75
>>> parse_timedelta_seconds('10:00:05')
36005
>>> parse_timedelta_seconds(12345)
12345
>>> parse_timedelta_seconds(datetime.timedelta(days=1, seconds=7))
86407
>>> parse_timedelta_seconds(1.1)
Traceback (most recent call last):
...
ValueError: Invalid duration (must be timedelta, int, or 'HH:MM:SS'): 1.1
>>> parse_timedelta_seconds("12:34:34bad")
Traceback (most recent call last):
...
ValueError: Invalid duration string, must be HH:MM:SS format
```

Definition at line 29 of file helpers.py.

5.7.1.12 `relative_or_absolute_path()`

```
def codar.cheetah.helpers.relative_or_absolute_path (
    prefix,
    path )
```

If path is an absolute path, return as is, otherwise pre-pend prefix.

Definition at line 130 of file helpers.py.

5.7.1.13 `relative_or_absolute_path_list()`

```
def codar.cheetah.helpers.relative_or_absolute_path_list (
    prefix,
    path_list )
```

Definition at line 137 of file helpers.py.

5.7.1.14 `require_campaign_directory()`

```
def codar.cheetah.helpers.require_campaign_directory (
    path )
```

Raise CheetahException if the specified path is not a top-level campaign directory.

Definition at line 193 of file helpers.py.

5.7.1.15 `swift_escape_string()`

```
def codar.cheetah.helpers.swift_escape_string (
    s )
```

Escape backslashes and double quotes in string, so it can be embedded in a literal swift string when generatig swift source code.

Definition at line 19 of file helpers.py.

5.8 codar.cheetah.launchers Namespace Reference

Classes

- class [Launcher](#)

Variables

- string `TAU_PROFILE_PATTERN` = "codar.cheetah.tau-{code}"

5.8.1 Detailed Description

Class model for "launchers", which are responsible for taking an application and mediating how it is run on a super computer or local machine. The only supported launcher currently is swift-t. Swift allows us to configure how each run within a sweep is parallelized, and handles details of submitting to the correct scheduler and runner when passed appropriate options.

5.8.2 Variable Documentation

5.8.2.1 TAU_PROFILE_PATTERN

```
string codar.cheetah.launchers.TAU_PROFILE_PATTERN = "codar.cheetah.tau-{code}"
```

Definition at line 25 of file launchers.py.

5.9 codar.cheetah.loader Namespace Reference

Functions

- def `load_experiment_class` (file_path)

5.9.1 Detailed Description

Functions for loading experiment python files by path.

Requires Python 3.3+

5.9.2 Function Documentation

5.9.2.1 load_experiment_class()

```
def codar.cheetah.loader.load_experiment_class (
    file_path )
```

Given the path to a python module containing an experiment, load the module and find and return the class.

Definition at line 12 of file loader.py.

5.10 codar.cheetah.machine_launchers Namespace Reference

Functions

- def [get_launcher](#) (machine, output_directory, num_codes)

Variables

- [machine_launchers](#) = dict()

5.10.1 Function Documentation

5.10.1.1 [get_launcher\(\)](#)

```
def codar.cheetah.machine_launchers.get_launcher (
    machine,
    output_directory,
    num_codes )
```

Definition at line 12 of file machine_launchers.py.

5.10.2 Variable Documentation

5.10.2.1 [machine_launchers](#)

```
codar.cheetah.machine_launchers.machine_launchers = dict()
```

Definition at line 4 of file machine_launchers.py.

5.11 codar.cheetah.model Namespace Reference

Classes

- class [Campaign](#)
- class [Run](#)
- class [RunComponent](#)

Variables

- [RESERVED_CODE_NAMES](#) = set(['post-process'])

5.11.1 Detailed Description

Object oriented model to represent jobs to run on different Supercomputers or workstations using different schedulers and runners (for running applications on compute nodes from front end nodes), and allow pass through of scheduler or runner specific options.

Subclasses representing specific types of schedulers, runners, and supercomputers (machines) are specified in other modules with the corresponding name.

5.11.2 Variable Documentation

5.11.2.1 RESERVED_CODE_NAMES

```
codar.cheetah.model.RESERVED_CODE_NAMES = set(['post-process'])
```

Definition at line 36 of file model.py.

5.12 codar.cheetah.parameters Namespace Reference

Classes

- class [CodeCommand](#)
- class [Instance](#)
- class [Param](#)
- class [ParamADIOS2XML](#)
- class [ParamAdiosXML](#)
- class [ParamCmdLineArg](#)
- class [ParamCmdLineOption](#)
- class [ParamConfig](#)
- class [ParamEnvVar](#)
- class [ParameterValue](#)
- class [ParamKeyValue](#)
- class [ParamRunner](#)
- class [ParamSchedulerArgs](#)
- class [SummitOpts](#)
- class [Sweep](#)
- class [SweepGroup](#)
- class [SymLink](#)

5.12.1 Detailed Description

Module containing classes for specifying parameter value sets and groupings of parameters. Used in the Experiment specification in the 'runs' variable.

5.13 codar.cheetah.pbs Namespace Reference

Functions

- def `open_pbs_file` (scheduler_dir_path, name, project, nodes, walltime)
- def `write_run_script` (script_out_path, scheduler_dir_path)

Variables

- string `PBS_NAME` = 'job.pbs'
- string `PBS_FORMAT_TEMPLATE`
- string `SUBMIT_FORMAT_TEMPLATE`

5.13.1 Detailed Description

Module for generating PBS files for executing many jobs with the same number of nodes.

TODO: define a common interface for schedulers, that works with at least SLURM and PBS.

TODO: codify dir structure - scheduler dir contains scheduler script, has subdir for each set of experiment parameters.

5.13.2 Function Documentation

5.13.2.1 `open_pbs_file()`

```
def codar.cheetah.pbs.open_pbs_file (  
    scheduler_dir_path,  
    name,  
    project,  
    nodes,  
    walltime )
```

Open and write a PBS file to the specified path and return the open file object for further writing. Caller is responsible for closing the file.

TODO: rather than passing back a file, this should probably return an object with an 'add_run' function. There should also be a template for the run output dir set somewhere - maybe other modules handle that, it should not be scheduler specific.

Definition at line 38 of file pbs.py.

5.13.2.2 write_run_script()

```
def codar.cheetah.pbs.write_run_script (
    script_out_path,
    scheduler_dir_path )
```

Write a bash script that will submit a PBS file generated by 'open_pbs_file' with the correct working directory and enironment. This is the end user (experiment runner)'s entry point to start the experiment.

Definition at line 55 of file pbs.py.

5.13.3 Variable Documentation

5.13.3.1 PBS_FORMAT_TEMPLATE

```
string codar.cheetah.pbs.PBS_FORMAT_TEMPLATE
```

Initial value:

```
1 = """
2 #!/bin/bash
3 #PBS -N {name}
4 #PBS -A {project}
5 #PBS -l nodes={nodes}
6 #PBS -l walltime={walltime}
7
8 """
```

Definition at line 18 of file pbs.py.

5.13.3.2 PBS_NAME

```
string codar.cheetah.pbs.PBS_NAME = 'job.pbs'
```

Definition at line 15 of file pbs.py.

5.13.3.3 SUBMIT_FORMAT_TEMPLATE

```
string codar.cheetah.pbs.SUBMIT_FORMAT_TEMPLATE
```

Initial value:

```
1 = """
2 #!/bin/bash
3
4 cd "{scheduler_directory}"
5 qsub {pbs_name}
6 """
```

Definition at line 30 of file pbs.py.

5.14 codar.cheetah.report_generator Namespace Reference

Classes

- class [_ReportGenerator](#)
- class [_RunParser](#)

Functions

- def [generate_report](#) (campaign_directory, user_run_script, output_file_path)

5.14.1 Detailed Description

Generate performance report from a completed campaign.
This module parses all run directories in all sweep groups to aggregate information.
Runs sosflow analysis to collect data.

All parameters specified in the spec file must be used as column headers in an output csv file.

5.14.2 Function Documentation

5.14.2.1 generate_report()

```
def codar.cheetah.report_generator.generate_report (
    campaign_directory,
    user_run_script,
    output_file_path )
```

This is a post-run function.
It walks the campaign tree and retrieves performance information about all completed runs.

Definition at line 401 of file report_generator.py.

5.15 codar.cheetah.runners Namespace Reference

Classes

- class [Runner](#)
- class [RunnerCray](#)
- class [RunnerLocal](#)

5.15.1 Detailed Description

TODO: unused currently by SwiftLauncher, but may still be needed, so keeping this module for now.

5.16 codar.cheetah.status Namespace Reference

Functions

- def [print_campaign_status](#) (campaign_directory, filter_user=None, filter_group=None, filter_run=None, filter_code=None, group_summary=False, run_summary=False, print_logs=False, log_level='DEBUG', return_codes=False, print_output=False, show_parameters=False)
- def [get_workflow_status](#) (status_file_path, print_counts=False, indent=0, print_return_codes=False, filter_run=None, print_parameters=False, filter_code=None, run_summary=False, code_names=None)

5.16.1 Detailed Description

Funtions to print status information for campaigns.

5.16.2 Function Documentation

5.16.2.1 [get_workflow_status\(\)](#)

```
def codar.cheetah.status.get_workflow_status (
    status_file_path,
    print_counts = False,
    indent = 0,
    print_return_codes = False,
    filter_run = None,
    print_parameters = False,
    filter_code = None,
    run_summary = False,
    code_names = None )
```

Definition at line 187 of file status.py.

5.16.2.2 [print_campaign_status\(\)](#)

```
def codar.cheetah.status.print_campaign_status (
    campaign_directory,
    filter_user = None,
    filter_group = None,
    filter_run = None,
    filter_code = None,
    group_summary = False,
    run_summary = False,
    print_logs = False,
    log_level = 'DEBUG',
    return_codes = False,
    print_output = False,
    show_parameters = False )
```

Definition at line 22 of file status.py.

5.17 codar.cheetah.templates Namespace Reference

Variables

- string [CAMPAIGN_ENV_TEMPLATE](#)
- string [GROUP_ENV_TEMPLATE](#)

5.17.1 Detailed Description

Templates for cheetah configuration. This should be used as little as possible: ideally scripts should be stored separately and be independently testable. For example, bash scripts can use environment variables for customization instead of being templates.

5.17.2 Variable Documentation

5.17.2.1 CAMPAIGN_ENV_TEMPLATE

string codar.cheetah.templates.CAMPAIGN_ENV_TEMPLATE

Initial value:

```
1 = """
2 export CODAR_CHEETAH_EXPERIMENT_DIR="{experiment_dir}"
3 export CODAR_CHEETAH_MACHINE_CONFIG="{machine_config}"
4 export CODAR_CHEETAH_APP_CONFIG="{app_config}"
5 export CODAR_WORKFLOW_SCRIPT="{workflow_script_path}"
6 export CODAR_WORKFLOW_RUNNER="{workflow_runner}"
7 export CODAR_CHEETAH_WORKFLOW_LOG_LEVEL="{workflow_debug_level}"
8 export CODAR_CHEETAH_UMASK="{umask}"
9 export CODAR_PYTHON="{codar_python}"
10 """
```

Definition at line 9 of file templates.py.

5.17.2.2 GROUP_ENV_TEMPLATE

string codar.cheetah.templates.GROUP_ENV_TEMPLATE

Initial value:

```
1 = """
2 export CODAR_CHEETAH_GROUP_WALLTIME="{walltime}"
3 export CODAR_CHEETAH_GROUP_MAX_PROCS="{max_procs}"
4
5 export CODAR_CHEETAH_SCHEDULER_ACCOUNT="{account}"
6 # queue on PBS, partition on SLURM
7 export CODAR_CHEETAH_SCHEDULER_QUEUE="{queue}"
8 # SLURM specific options
9 export CODAR_CHEETAH_SCHEDULER_CONSTRAINT="{constraint}"
10 export CODAR_CHEETAH_SCHEDULER_LICENSE="{license}"
11
12 export CODAR_CHEETAH_CAMPAIGN_NAME="{campaign_name}"
13
14 export CODAR_CHEETAH_GROUP_NAME="{group_name}"
15 export CODAR_CHEETAH_GROUP_NODES="{nodes}"
16 export CODAR_CHEETAH_GROUP_NODE_EXCLUSIVE="{node_exclusive}"
17 export CODAR_CHEETAH_GROUP_PROCESSES_PER_NODE="{processes_per_node}"
18 export CODAR_CHEETAH_MACHINE_NAME="{machine_name}"
19 """
```

Definition at line 21 of file templates.py.

5.18 codar.savanna Namespace Reference

Namespaces

- [consumer](#)
- [exc](#)
- [machines](#)
- [main](#)
- [model](#)
- [node_layout](#)
- [producer](#)
- [runners](#)
- [scheduler](#)
- [status](#)
- [submit_helper](#)

5.18.1 Detailed Description

Classes for running pipelines of MPI tasks based on a specified total process limit. The system is designed to use two + N threads:

1. consumer thread: get pipelines from queue and execute them when process slots become available. Stops when a None pipeline is received.
2. producer thread: add pipelines to queue. Can be from file or from network service.
3. monitor threads: each process spawned by the consumer thread has a monitor thread that blocks on the processes completing with a timeout, and kills the process if it's not done after the timeout is reached.

5.19 codar.savanna.consumer Namespace Reference

Classes

- class [PipelineRunner](#)

5.19.1 Detailed Description

Classes for 'consuming' pipelines - running groups of MPI tasks based on a specified total process limit.

5.20 codar.savanna.exc Namespace Reference

Classes

- class [MachineNotFound](#)
- class [SavannaException](#)

5.20.1 Detailed Description

Exceptions.

5.21 codar.savanna.machines Namespace Reference

Classes

- class [Machine](#)
- class [MachineNode](#)
- class [SummitNode](#)

Functions

- def [get_by_name](#) (name)

Variables

- [SCHEDULER_OPTIONS](#) = set(["project", "queue", "constraint", "license"])
- [local](#) = [Machine](#)('local', "local", "mpiexec", MachineNode, processes_per_node=1)
- [titan](#)
- [cori](#)
- [theta](#)
- [summit](#)

5.21.1 Detailed Description

Configuration for machines supported by Codar.

5.21.2 Function Documentation

5.21.2.1 [get_by_name\(\)](#)

```
def codar.savanna.machines.get_by_name (
    name )
```

Definition at line 149 of file machines.py.

5.21.3 Variable Documentation

5.21.3.1 cori

`codar.savanna.machines.cori`

Initial value:

```
1 = Machine('cori', "slurm", "srun", MachineNode,
2           processes_per_node=32, node_exclusive=True,
3           dataspace_servers_per_node=4,
4           scheduler_options=dict(project="",
5                                   queue="debug",
6                                   constraint="haswell",
7                                   license="SCRATCH,project"))
```

Definition at line 128 of file machines.py.

5.21.3.2 local

`codar.savanna.machines.local = Machine('local', "local", "mpiexec", MachineNode, processes_per_node=1)`

Definition at line 118 of file machines.py.

5.21.3.3 SCHEDULER_OPTIONS

`codar.savanna.machines.SCHEDULER_OPTIONS = set(["project", "queue", "constraint", "license"])`

Definition at line 13 of file machines.py.

5.21.3.4 summit

`codar.savanna.machines.summit`

Initial value:

```
1 = Machine('summit', "ibm_lsf", "jsrun", SummitNode,
2           processes_per_node=42, node_exclusive=True,
3           scheduler_options=dict(project=""))
```

Definition at line 144 of file machines.py.

5.21.3.5 theta

`codar.savanna.machines.theta`

Initial value:

```
1 = Machine('theta', "cobalt", "aprun", MachineNode,
2           processes_per_node=64, node_exclusive=True,
3           dataspace_servers_per_node=8,
4           scheduler_options=dict(project="",
5                                   queue="debug-flat-quad"))
```

Definition at line 137 of file machines.py.

5.21.3.6 titan

`codar.savanna.machines.titan`

Initial value:

```
1 = Machine('titan', "pbs", "aprun", MachineNode,
2           processes_per_node=16, node_exclusive=True,
3           scheduler_options=dict(project="", queue="debug"),
4           dataspace_servers_per_node=4)
```

Definition at line 120 of file machines.py.

5.22 codar.savanna.main Namespace Reference

Functions

- def [parse_args](#) ()
- def [main](#) ()
- def [get_job_id](#) ()

Variables

- [consumer](#) = None

5.22.1 Detailed Description

Main program for executing workflow script with different producers and runners.

5.22.2 Function Documentation

5.22.2.1 `get_job_id()`

```
def codar.savanna.main.get_job_id ( )
```

Definition at line 104 of file main.py.

5.22.2.2 `main()`

```
def codar.savanna.main.main ( )
```

Definition at line 39 of file main.py.

5.22.2.3 `parse_args()`

```
def codar.savanna.main.parse_args ( )
```

Definition at line 18 of file main.py.

5.22.3 Variable Documentation

5.22.3.1 `consumer`

```
codar.savanna.main.consumer = None
```

Definition at line 15 of file main.py.

5.23 `codar.savanna.model` Namespace Reference

Classes

- class [NodeConfig](#)
- class [Pipeline](#)
- class [Run](#)

Variables

- string [STDOUT_NAME](#) = 'codar.workflow.stdout'
- string [STDERR_NAME](#) = 'codar.workflow.stderr'
- string [RETURN_NAME](#) = 'codar.workflow.return'
- string [WALLTIME_NAME](#) = 'codar.workflow.walltime'
- int [KILL_WAIT](#) = 30
- int [WAIT_DELAY_KILL](#) = 30
- int [WAIT_DELAY_GIVE_UP](#) = 120

5.23.1 Detailed Description

Classes for tracking pipelines and the runs within each pipeline in separate monitor threads that synchronize state.

Note that there is state tracked in these classes which is not available just by looking at the return code. In particular, a run may be killed for several different reasons: external signal, run timeout reached, other run in pipeline failed (when kill on partial fail is set), or if the entire workflow is killed.

The goal here is to provide as much information as possible about why a pipeline failed, to make an informed decision about whether it is worth running again when the workflow is restarted, or if its failure was more permanent and not subject to outside forces like the job walltime expiring.

5.23.2 Variable Documentation

5.23.2.1 KILL_WAIT

```
int codar.savanna.model.KILL_WAIT = 30
```

Definition at line 37 of file model.py.

5.23.2.2 RETURN_NAME

```
string codar.savanna.model.RETURN_NAME = 'codar.workflow.return'
```

Definition at line 34 of file model.py.

5.23.2.3 STDERR_NAME

```
string codar.savanna.model.STDERR_NAME = 'codar.workflow.stderr'
```

Definition at line 33 of file model.py.

5.23.2.4 STDOUT_NAME

```
string codar.savanna.model.STDOUT_NAME = 'codar.workflow.stdout'
```

Definition at line 32 of file model.py.

5.23.2.5 WAIT_DELAY_GIVE_UP

```
int codar.savanna.model.WAIT_DELAY_GIVE_UP = 120
```

Definition at line 39 of file model.py.

5.23.2.6 WAIT_DELAY_KILL

```
int codar.savanna.model.WAIT_DELAY_KILL = 30
```

Definition at line 38 of file model.py.

5.23.2.7 WALLTIME_NAME

```
string codar.savanna.model.WALLTIME_NAME = 'codar.workflow.walltime'
```

Definition at line 35 of file model.py.

5.24 codar.savanna.node_layout Namespace Reference

Classes

- class [NodeLayout](#)

5.25 codar.savanna.producer Namespace Reference

Classes

- class [JSONFilePipelineReader](#)

5.25.1 Detailed Description

Classes for producing pipelines.

5.26 codar.savanna.runners Namespace Reference

Classes

- class [MPIRunner](#)
- class [Runner](#)
- class [SummitRunner](#)

Variables

- `mpiexec` = `MPIRunner('mpiexec', '-n', hostfile='--hostfile')`
- `aprun` = `MPIRunner('aprun', '-n', tasks_per_node_arg='-N', hostfile='-L')`
- `srun` = `MPIRunner('srun', '-n', nodes_arg='-N', hostfile='-w')`
- `jsrun` = `SummitRunner()`

5.26.1 Variable Documentation

5.26.1.1 aprun

```
codar.savanna.runners.aprun = MPIRunner('aprun', '-n', tasks_per_node_arg='-N', hostfile='-L')
```

Definition at line 94 of file runners.py.

5.26.1.2 jsrun

```
codar.savanna.runners.jsrun = SummitRunner()
```

Definition at line 96 of file runners.py.

5.26.1.3 mpiexec

```
codar.savanna.runners.mpiexec = MPIRunner('mpiexec', '-n', hostfile='--hostfile')
```

Definition at line 93 of file runners.py.

5.26.1.4 srun

```
codar.savanna.runners.srun = MPIRunner('srun', '-n', nodes_arg='-N', hostfile='-w')
```

Definition at line 95 of file runners.py.

5.27 codar.savanna.scheduler Namespace Reference

Classes

- class `JobList`

5.27.1 Detailed Description

Classes related to finding a job that can run on available resources. Does not assume any knowledge of how long each job will take. Designed for greedy search of a job that will fit whenever resources are freed.

In the context of Cheetah workflows, it's unlikely that there will be more than a few hundred jobs, so it's not worth optimizing the python search code very much. It is however worth making sure that a job is run when resources are available, since super computer resources are expensive. Basically it's worth doing some work in python to make sure we start a big unit of work on compute nodes.

5.28 codar.savanna.status Namespace Reference

Classes

- class [PipelineState](#)
- class [WorkflowStatus](#)

Variables

- string [NOT_STARTED](#) = 'not_started'
- string [RUNNING](#) = 'running'
- string [DONE](#) = 'done'
- string [KILLED](#) = 'killed'
- string [REASON_TIMEOUT](#) = 'timeout'
- string [REASON_FAILED](#) = 'failed'
- string [REASON_SUCCEEDED](#) = 'succeeded'
- string [REASON_EXCEPTION](#) = 'exception'
- string [REASON_NOFIT](#) = 'nofit'

5.28.1 Detailed Description

Class for maintaining state of all FOB runs that the workflow consumer is managing. State is saved in a JSON file, overwritten on each state change.

5.28.2 Variable Documentation

5.28.2.1 DONE

```
string codar.savanna.status.DONE = 'done'
```

Definition at line 14 of file status.py.

5.28.2.2 KILLED

```
string codar.savanna.status.KILLED = 'killed'
```

Definition at line 15 of file status.py.

5.28.2.3 NOT_STARTED

```
string codar.savanna.status.NOT_STARTED = 'not_started'
```

Definition at line 12 of file status.py.

5.28.2.4 REASON_EXCEPTION

```
string codar.savanna.status.REASON_EXCEPTION = 'exception'
```

Definition at line 20 of file status.py.

5.28.2.5 REASON_FAILED

```
string codar.savanna.status.REASON_FAILED = 'failed'
```

Definition at line 18 of file status.py.

5.28.2.6 REASON_NOFIT

```
string codar.savanna.status.REASON_NOFIT = 'nofit'
```

Definition at line 21 of file status.py.

5.28.2.7 REASON_SUCCEEDED

```
string codar.savanna.status.REASON_SUCCEEDED = 'succeeded'
```

Definition at line 19 of file status.py.

5.28.2.8 REASON_TIMEOUT

```
string codar.savanna.status.REASON_TIMEOUT = 'timeout'
```

Definition at line 17 of file status.py.

5.28.2.9 RUNNING

```
string codar.savanna.status.RUNNING = 'running'
```

Definition at line 13 of file status.py.

5.29 codar.savanna.summit_helper Namespace Reference

Functions

- def [get_nodes_reqd](#) (res_set, nrs)
- def [create_erf_file](#) (run)

5.29.1 Function Documentation

5.29.1.1 create_erf_file()

```
def codar.savanna.summit_helper.create_erf_file (  
    run )
```

Definition at line 12 of file summit_helper.py.

5.29.1.2 get_nodes_reqd()

```
def codar.savanna.summit_helper.get_nodes_reqd (  
    res_set,  
    nrs )
```

Get the no. of nodes that will be required based on the resource set
and the no. of resource sets

Definition at line 5 of file summit_helper.py.

Chapter 6

Class Documentation

6.1 codar.cheetah.report_generator._ReportGenerator Class Reference

Public Member Functions

- def `__init__` (self, `campaign_directory`, `user_run_script`, `output_filename`)
- def `parse_campaign` (self)
- def `parse_user_campaigns` (self)
- def `parse_sweep_group` (self, `group_dir`)
- def `parse_run_dir` (self, `run_dir`, `exit_status`)
- def `write_output` (self)

Public Attributes

- `parsed_runs`
- `unique_keys`
- `campaign_directory`
- `user_run_script`
- `output_filename`
- `current_campaign_user`
- `run_status`

6.1.1 Detailed Description

Definition at line 228 of file `report_generator.py`.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 `__init__()`

```
def codar.cheetah.report_generator._ReportGenerator.__init__ (
    self,
    campaign_directory,
    user_run_script,
    output_filename )
```

Definition at line 232 of file `report_generator.py`.

6.1.3 Member Function Documentation

6.1.3.1 `parse_campaign()`

```
def codar.cheetah.report_generator._ReportGenerator.parse_campaign (
    self )

: return:
```

Definition at line 259 of file `report_generator.py`.

6.1.3.2 `parse_run_dir()`

```
def codar.cheetah.report_generator._ReportGenerator.parse_run_dir (
    self,
    run_dir,
    exit_status )
```

Parse run directory of a sweep group

Definition at line 331 of file `report_generator.py`.

6.1.3.3 `parse_sweep_group()`

```
def codar.cheetah.report_generator._ReportGenerator.parse_sweep_group (
    self,
    group_dir )
```

Parse sweep group and get post-run performance information

Definition at line 304 of file `report_generator.py`.

6.1.3.4 parse_user_campaigns()

```
def codar.cheetah.report_generator._ReportGenerator.parse_user_campaigns (
    self )

: return:
```

Definition at line 273 of file report_generator.py.

6.1.3.5 write_output()

```
def codar.cheetah.report_generator._ReportGenerator.write_output (
    self )

: return:
```

Definition at line 388 of file report_generator.py.

6.1.4 Member Data Documentation

6.1.4.1 campaign_directory

```
codar.cheetah.report_generator._ReportGenerator.campaign_directory
```

Definition at line 241 of file report_generator.py.

6.1.4.2 current_campaign_user

```
codar.cheetah.report_generator._ReportGenerator.current_campaign_user
```

Definition at line 254 of file report_generator.py.

6.1.4.3 output_filename

```
codar.cheetah.report_generator._ReportGenerator.output_filename
```

Definition at line 251 of file report_generator.py.

6.1.4.4 parsed_runs

`codar.cheetah.report_generator._ReportGenerator.parsed_runs`

Definition at line 235 of file `report_generator.py`.

6.1.4.5 run_status

`codar.cheetah.report_generator._ReportGenerator.run_status`

Definition at line 257 of file `report_generator.py`.

6.1.4.6 unique_keys

`codar.cheetah.report_generator._ReportGenerator.unique_keys`

Definition at line 239 of file `report_generator.py`.

6.1.4.7 user_run_script

`codar.cheetah.report_generator._ReportGenerator.user_run_script`

Definition at line 247 of file `report_generator.py`.

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

- [cheetah/report_generator.py](#)

6.2 `codar.cheetah.report_generator._RunParser` Class Reference

Public Member Functions

- `def __init__ (self, run_dir, exit_status, user_run_script)`
- `def read_fob_json (self)`
- `def get_rc_names (self)`
- `def get_run_params (self)`
- `def read_sos_perf_data (self)`
- `def get_cheetah_perf_data (self)`
- `def read_adios_output_file_sizes (self)`
- `def read_node_layout (self)`
- `def execute_user_run_script (self)`
- `def verify_run_successful (self)`
- `def serialize_params_nested_dict (self, nested_run_params_dict)`

Public Attributes

- [run_dir](#)
- [exit_status](#)
- [user_run_script](#)
- [serialized_run_params](#)
- [fob_dict](#)
- [rc_names](#)
- [rc_working_dir](#)
- [rc_name_exe](#)

6.2.1 Detailed Description

Definition at line 22 of file report_generator.py.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 __init__()

```
def codar.cheetah.report_generator._RunParser.__init__ (
    self,
    run_dir,
    exit_status,
    user_run_script )
```

Class to parse a run directory.
:param run_dir:

Definition at line 23 of file report_generator.py.

6.2.3 Member Function Documentation

6.2.3.1 execute_user_run_script()

```
def codar.cheetah.report_generator._RunParser.execute_user_run_script (
    self )
```

Definition at line 160 of file report_generator.py.

6.2.3.2 `get_cheetah_perf_data()`

```
def codar.cheetah.report_generator._RunParser.get_cheetah_perf_data (
    self )
```

Definition at line 115 of file `report_generator.py`.

6.2.3.3 `get_rc_names()`

```
def codar.cheetah.report_generator._RunParser.get_rc_names (
    self )
```

Definition at line 52 of file `report_generator.py`.

6.2.3.4 `get_run_params()`

```
def codar.cheetah.report_generator._RunParser.get_run_params (
    self )
```

Definition at line 70 of file `report_generator.py`.

6.2.3.5 `read_adios_output_file_sizes()`

```
def codar.cheetah.report_generator._RunParser.read_adios_output_file_sizes (
    self )
```

```
:return:
```

Definition at line 127 of file `report_generator.py`.

6.2.3.6 `read_fob_json()`

```
def codar.cheetah.report_generator._RunParser.read_fob_json (
    self )
```

Definition at line 46 of file `report_generator.py`.

6.2.3.7 read_node_layout()

```
def codar.cheetah.report_generator._RunParser.read_node_layout (
    self )

:return:
```

Definition at line 149 of file report_generator.py.

6.2.3.8 read_sos_perf_data()

```
def codar.cheetah.report_generator._RunParser.read_sos_perf_data (
    self )

:return: True if sos data was found, False otherwise
```

Definition at line 82 of file report_generator.py.

6.2.3.9 serialize_params_nested_dict()

```
def codar.cheetah.report_generator._RunParser.serialize_params_nested_dict (
    self,
    nested_run_params_dict )
```

codar.cheetah.run-params.json has the structure:

```
{
  app1: {
    param1: value1
    param2: value2
  }
  app2: {
    param1: value1
    param2: value2
  }
}
```

Serialize this structure so that we have
{app1__param1: value1, app1__param2:value2, and so on}.

Definition at line 204 of file report_generator.py.

6.2.3.10 verify_run_successful()

```
def codar.cheetah.report_generator._RunParser.verify_run_successful (
    self )

:return:
```

Definition at line 180 of file report_generator.py.

6.2.4 Member Data Documentation

6.2.4.1 `exit_status`

`codar.cheetah.report_generator._RunParser.exit_status`

Definition at line 30 of file `report_generator.py`.

6.2.4.2 `fob_dict`

`codar.cheetah.report_generator._RunParser.fob_dict`

Definition at line 34 of file `report_generator.py`.

6.2.4.3 `rc_name_exe`

`codar.cheetah.report_generator._RunParser.rc_name_exe`

Definition at line 41 of file `report_generator.py`.

6.2.4.4 `rc_names`

`codar.cheetah.report_generator._RunParser.rc_names`

Definition at line 35 of file `report_generator.py`.

6.2.4.5 `rc_working_dir`

`codar.cheetah.report_generator._RunParser.rc_working_dir`

Definition at line 36 of file `report_generator.py`.

6.2.4.6 run_dir

```
codar.cheetah.report_generator._RunParser.run_dir
```

Definition at line 29 of file report_generator.py.

6.2.4.7 serialized_run_params

```
codar.cheetah.report_generator._RunParser.serialized_run_params
```

Definition at line 33 of file report_generator.py.

6.2.4.8 user_run_script

```
codar.cheetah.report_generator._RunParser.user_run_script
```

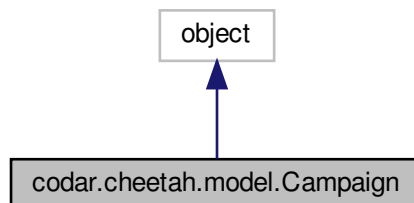
Definition at line 31 of file report_generator.py.

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

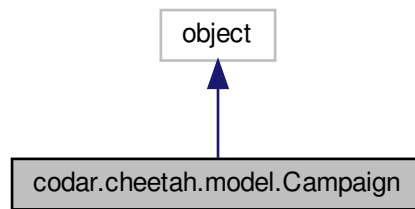
- [cheetah/report_generator.py](#)

6.3 codar.cheetah.model.Campaign Class Reference

Inheritance diagram for codar.cheetah.model.Campaign:



Collaboration diagram for `codar.cheetah.model.Campaign`:



Public Member Functions

- `def __init__` (self, machine_name, [app_dir](#))
- `def make_experiment_run_dir` (self, output_dir, _check_code_paths=True)

Public Attributes

- [machine](#)
- [app_dir](#)
- [runs](#)
- [inputs](#)
- [codes](#)
- [machine_scheduler_options](#)
- [machine_app_config_script](#)

Static Public Attributes

- [name](#) = None
- list [codes](#) = []
- list [supported_machines](#) = []
- list [sweeps](#) = []
- list [inputs](#) = []
- [umask](#) = None
- bool [kill_on_partial_failure](#) = False
- [run_post_process_script](#) = None
- bool [run_post_process_stop_group_on_failure](#) = False
- [app_config_scripts](#) = None
- [run_dir_setup_script](#) = None
- dictionary [scheduler_options](#) = {}
- [tau_config](#) = None
- [sosd_path](#) = None
- [sos_analysis_path](#) = None
- int [sosd_num_aggregators](#) = 1
- [post_process_script](#) = None
- [python_path](#) = `sys.executable`

6.3.1 Detailed Description

An experiment class specifies an application, a set of parameter to sweep over, and a set of supported target machine. A specific instance binds the experiment to a specific machine within the set of supported machines, and supports generating a set of scripts to run the experiment on that machine.

Definition at line 39 of file model.py.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 `__init__()`

```
def codar.cheetah.model.Campaign.__init__ (
    self,
    machine_name,
    app_dir )
```

Definition at line 117 of file model.py.

6.3.3 Member Function Documentation

6.3.3.1 `make_experiment_run_dir()`

```
def codar.cheetah.model.Campaign.make_experiment_run_dir (
    self,
    output_dir,
    _check_code_paths = True )
```

Produce scripts and directory structure for running the experiment.

Directory structure will be a subdirectory for each scheduler group, and within each scheduler group directory, a subdirectory for each run.

Definition at line 199 of file model.py.

6.3.4 Member Data Documentation

6.3.4.1 app_config_scripts

```
codar.cheetah.model.Campaign.app_config_scripts = None [static]
```

Definition at line 74 of file model.py.

6.3.4.2 app_dir

```
codar.cheetah.model.Campaign.app_dir
```

Definition at line 126 of file model.py.

6.3.4.3 codes [1/2]

```
list codar.cheetah.model.Campaign.codes = [] [static]
```

Definition at line 48 of file model.py.

6.3.4.4 codes [2/2]

```
codar.cheetah.model.Campaign.codes
```

Definition at line 134 of file model.py.

6.3.4.5 inputs [1/2]

```
list codar.cheetah.model.Campaign.inputs = [] [static]
```

Definition at line 51 of file model.py.

6.3.4.6 inputs [2/2]

```
codar.cheetah.model.Campaign.inputs
```

Definition at line 131 of file model.py.

6.3.4.7 kill_on_partial_failure

```
bool codar.cheetah.model.Campaign.kill_on_partial_failure = False [static]
```

Definition at line 56 of file model.py.

6.3.4.8 machine

```
codar.cheetah.model.Campaign.machine
```

Definition at line 125 of file model.py.

6.3.4.9 machine_app_config_script

```
codar.cheetah.model.Campaign.machine_app_config_script
```

Definition at line 180 of file model.py.

6.3.4.10 machine_scheduler_options

```
codar.cheetah.model.Campaign.machine_scheduler_options
```

Definition at line 174 of file model.py.

6.3.4.11 name

```
codar.cheetah.model.Campaign.name = None [static]
```

Definition at line 47 of file model.py.

6.3.4.12 post_process_script

```
codar.cheetah.model.Campaign.post_process_script = None [static]
```

Definition at line 106 of file model.py.

6.3.4.13 python_path

```
codar.cheetah.model.Campaign.python_path = sys.executable [static]
```

Definition at line 112 of file model.py.

6.3.4.14 run_dir_setup_script

```
codar.cheetah.model.Campaign.run_dir_setup_script = None [static]
```

Definition at line 83 of file model.py.

6.3.4.15 run_post_process_script

```
codar.cheetah.model.Campaign.run_post_process_script = None [static]
```

Definition at line 64 of file model.py.

6.3.4.16 run_post_process_stop_group_on_failure

```
bool codar.cheetah.model.Campaign.run_post_process_stop_group_on_failure = False [static]
```

Definition at line 65 of file model.py.

6.3.4.17 runs

```
codar.cheetah.model.Campaign.runs
```

Definition at line 127 of file model.py.

6.3.4.18 scheduler_options

```
dictionary codar.cheetah.model.Campaign.scheduler_options = {} [static]
```

Definition at line 87 of file model.py.

6.3.4.19 sos_analysis_path

```
codar.cheetah.model.Campaign.sos_analysis_path = None [static]
```

Definition at line 96 of file model.py.

6.3.4.20 sosd_num_aggregators

```
int codar.cheetah.model.Campaign.sosd_num_aggregators = 1 [static]
```

Definition at line 97 of file model.py.

6.3.4.21 sosd_path

```
codar.cheetah.model.Campaign.sosd_path = None [static]
```

Definition at line 95 of file model.py.

6.3.4.22 supported_machines

```
list codar.cheetah.model.Campaign.supported_machines = [] [static]
```

Definition at line 49 of file model.py.

6.3.4.23 sweeps

```
list codar.cheetah.model.Campaign.sweeps = [] [static]
```

Definition at line 50 of file model.py.

6.3.4.24 tau_config

```
codar.cheetah.model.Campaign.tau_config = None [static]
```

Definition at line 90 of file model.py.

6.3.4.25 umask

```
codar.cheetah.model.Campaign.umask = None [static]
```

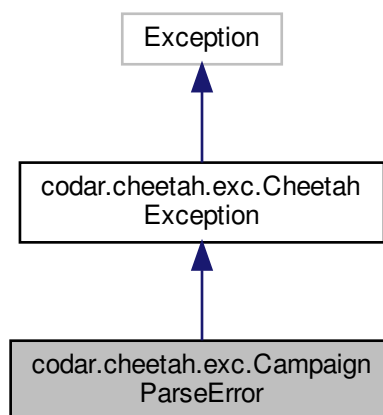
Definition at line 52 of file model.py.

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

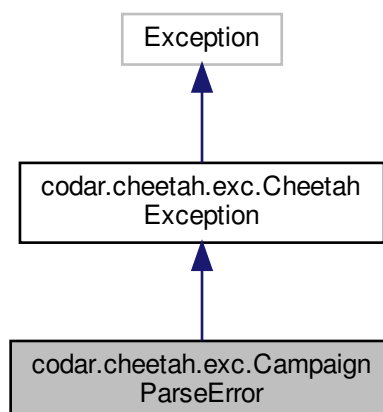
- [cheetah/model.py](#)

6.4 codar.cheetah.exc.CampaignParseError Class Reference

Inheritance diagram for codar.cheetah.exc.CampaignParseError:



Collaboration diagram for codar.cheetah.exc.CampaignParseError:



6.4.1 Detailed Description

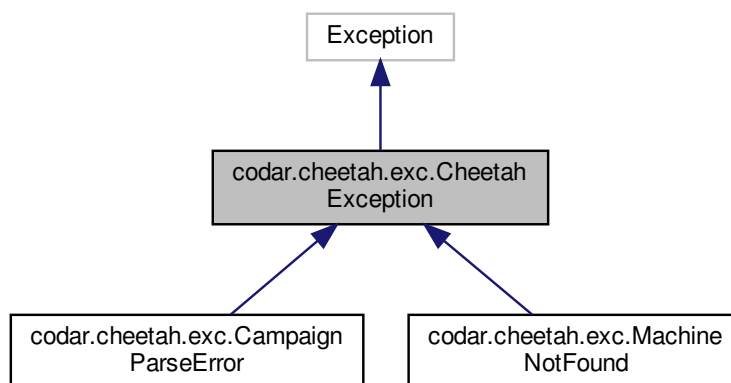
Definition at line 16 of file exc.py.

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

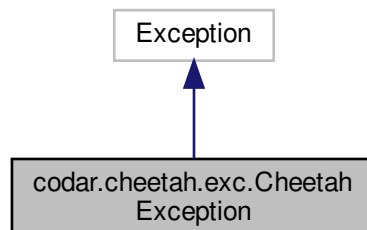
- [cheetah/exc.py](#)

6.5 codar.cheetah.exc.CheetahException Class Reference

Inheritance diagram for codar.cheetah.exc.CheetahException:



Collaboration diagram for codar.cheetah.exc.CheetahException:



6.5.1 Detailed Description

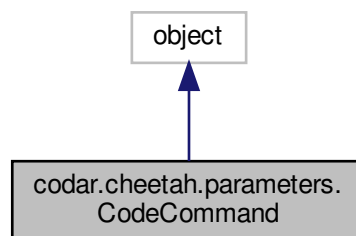
Definition at line 6 of file exc.py.

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

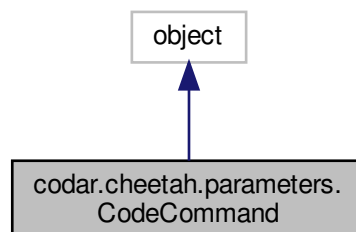
- cheetah/[exc.py](#)

6.6 `codar.cheetah.parameters.CodeCommand` Class Reference

Inheritance diagram for `codar.cheetah.parameters.CodeCommand`:



Collaboration diagram for `codar.cheetah.parameters.CodeCommand`:



Public Member Functions

- def [__init__](#) (self, [target](#))
- def [add_arg](#) (self, position, value)
- def [add_option](#) (self, option, value)
- def [get_argv](#) (self)

Public Attributes

- [target](#)
- [args](#)
- [options](#)

6.6.1 Detailed Description

Helper class to build up command args and options as we go. Does not know about the path to it's executable, that is part of the execution environment which is added during realization.

Definition at line 254 of file parameters.py.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 `__init__()`

```
def codar.cheetah.parameters.CodeCommand.__init__ (
    self,
    target )
```

Definition at line 260 of file parameters.py.

6.6.3 Member Function Documentation

6.6.3.1 `add_arg()`

```
def codar.cheetah.parameters.CodeCommand.add_arg (
    self,
    position,
    value )
```

Allows adding positional args out of order.

TODO: better error handling.

Definition at line 265 of file parameters.py.

6.6.3.2 `add_option()`

```
def codar.cheetah.parameters.CodeCommand.add_option (
    self,
    option,
    value )
```

Definition at line 277 of file parameters.py.

6.6.3.3 `get_argv()`

```
def codar.cheetah.parameters.CodeCommand.get_argv (
    self )
```

Definition at line 282 of file parameters.py.

6.6.4 Member Data Documentation

6.6.4.1 `args`

```
codar.cheetah.parameters.CodeCommand.args
```

Definition at line 262 of file parameters.py.

6.6.4.2 `options`

```
codar.cheetah.parameters.CodeCommand.options
```

Definition at line 263 of file parameters.py.

6.6.4.3 `target`

```
codar.cheetah.parameters.CodeCommand.target
```

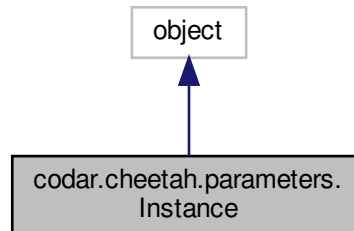
Definition at line 261 of file parameters.py.

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

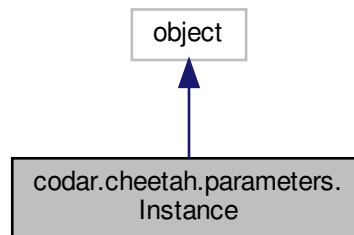
- cheetah/[parameters.py](#)

6.7 codar.cheetah.parameters.Instance Class Reference

Inheritance diagram for codar.cheetah.parameters.Instance:



Collaboration diagram for codar.cheetah.parameters.Instance:



Public Member Functions

- def [__init__](#) (self)
- def [add_parameter](#) (self, p, idx)
- def [parameter_values](#) (self)
- def [code_commands](#) (self)
- def [get_codes_argv](#) (self)
- def [as_string](#) (self)
- def [get_parameter_values_by_type](#) (self, param_class)
- def [get_nprocs](#) (self, target)
- def [get_hostfile](#) (self, target)
- def [get_sched_opts](#) (self, target)
- def [as_dict](#) (self)

6.7.1 Detailed Description

Represent an instance of an application with fixed parameters. An application may consist of multiple codes running at the same time, and multiple middleware layers (scheduler like PBS, runner like aprun, or swift), all of which may have their own parameters.

Abstractly, an instance is a two-level nested dict, where the first level indicates the target for a parameter (application code or middleware), and the second level contains the parameter values for that target.

Definition at line 101 of file parameters.py.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 `__init__()`

```
def codar.cheetah.parameters.Instance.__init__ (
    self )
```

Definition at line 113 of file parameters.py.

6.7.3 Member Function Documentation

6.7.3.1 `add_parameter()`

```
def codar.cheetah.parameters.Instance.add_parameter (
    self,
    p,
    idx )
```

Definition at line 129 of file parameters.py.

6.7.3.2 `as_dict()`

```
def codar.cheetah.parameters.Instance.as_dict (
    self )
```

Produce dict (mainly for for JSON serialization) with keys based on parameter names. This ignores the type of the param, it's just the name value pairs.

Definition at line 244 of file parameters.py.

6.7.3.3 as_string()

```
def codar.cheetah.parameters.Instance.as_string (
    self )
```

Get a command line like value for the instance. Note that this only includes positional and option command line args, not config args like adios XML. TODO: deprecate??

Definition at line 199 of file parameters.py.

6.7.3.4 code_commands()

```
def codar.cheetah.parameters.Instance.code_commands (
    self )
```

Wrapper to allow delayed calculation of derived parameter values.

Definition at line 146 of file parameters.py.

6.7.3.5 get_codes_argv()

```
def codar.cheetah.parameters.Instance.get_codes_argv (
    self )
```

Get an `_unordered_dict` mapping code name to list of args for that code. Higher levels of model are responsible for re-ordering as needed.

Definition at line 192 of file parameters.py.

6.7.3.6 get_hostfile()

```
def codar.cheetah.parameters.Instance.get_hostfile (
    self,
    target )
```

Definition at line 232 of file parameters.py.

6.7.3.7 `get_nprocs()`

```
def codar.cheetah.parameters.Instance.get_nprocs (
    self,
    target )
```

Definition at line 226 of file parameters.py.

6.7.3.8 `get_parameter_values_by_type()`

```
def codar.cheetah.parameters.Instance.get_parameter_values_by_type (
    self,
    param_class )
```

Get a list of ParamaterValues of the specified type in the instance.

Definition at line 215 of file parameters.py.

6.7.3.9 `get_sched_opts()`

```
def codar.cheetah.parameters.Instance.get_sched_opts (
    self,
    target )
```

Definition at line 238 of file parameters.py.

6.7.3.10 `parameter_values()`

```
def codar.cheetah.parameters.Instance.parameter_values (
    self )
```

Wrapper to allow delayed calculation of derived parameter values.

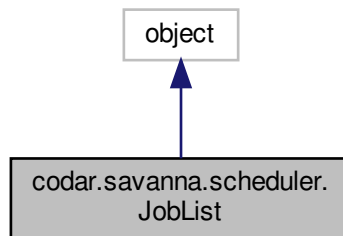
Definition at line 139 of file parameters.py.

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

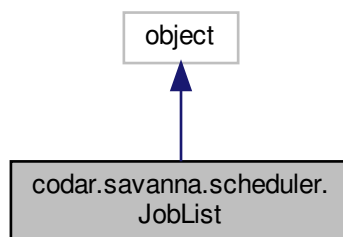
- cheetah/[parameters.py](#)

6.8 codar.savanna.scheduler.JobList Class Reference

Inheritance diagram for codar.savanna.scheduler.JobList:



Collaboration diagram for codar.savanna.scheduler.JobList:



Public Member Functions

- def `__init__` (self, costfn, initial_jobs=None)
- def `add_job` (self, job)
- def `pop_job` (self, max_cost)
- def `__len__` (self)

6.8.1 Detailed Description

Manage a job list that can find and remove the highest cost job that doesn't exceed max_cost and insert new jobs.

The job objects can be any type, but a key function must be provided that takes an instance of a job and returns it's cost.

Uses a coordinated pair of sort list for costs and jobs, along with the bisect module. A linked list might be more efficient, since the list copy on insert and delete may dominate the time to do a linear search of a small list, but it's likely fine either way for the sizes we will encounter.

Definition at line 18 of file scheduler.py.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 `__init__()`

```
def codar.savanna.scheduler.JobList.__init__ (
    self,
    costfn,
    initial_jobs = None )
```

Definition at line 30 of file scheduler.py.

6.8.3 Member Function Documentation

6.8.3.1 `__len__()`

```
def codar.savanna.scheduler.JobList.__len__ (
    self )
```

Definition at line 63 of file scheduler.py.

6.8.3.2 `add_job()`

```
def codar.savanna.scheduler.JobList.add_job (
    self,
    job )
```

Definition at line 41 of file scheduler.py.

6.8.3.3 `pop_job()`

```
def codar.savanna.scheduler.JobList.pop_job (
    self,
    max_cost )
```

Get the highest cost job that doesn't exceed `max_cost`, and remove it from the job list. Raises `IndexError` if the job list is empty, returns `None` if no suitable jobs exist in the list.

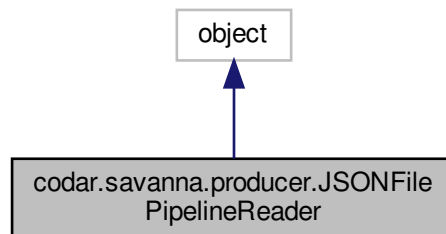
Definition at line 48 of file scheduler.py.

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

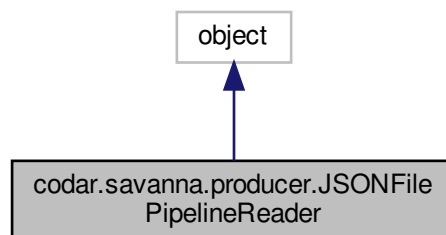
- savanna/[scheduler.py](#)

6.9 codar.savanna.producer.JSONFilePipelineReader Class Reference

Inheritance diagram for codar.savanna.producer.JSONFilePipelineReader:



Collaboration diagram for codar.savanna.producer.JSONFilePipelineReader:



Public Member Functions

- `def __init__(self, file_path)`
- `def read_pipelines(self)`

Public Attributes

- `file_path`

6.9.1 Detailed Description

Load pipelines from a file formatted as a new line separated list of JSON documents. Each JSON document must be a list containing dictionaries, each dictionary describing a code to run as part of the pipeline.

Definition at line 12 of file producer.py.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 `__init__()`

```
def codar.savanna.producer.JSONFilePipelineReader.__init__ (
    self,
    file_path )
```

Definition at line 17 of file producer.py.

6.9.3 Member Function Documentation

6.9.3.1 `read_pipelines()`

```
def codar.savanna.producer.JSONFilePipelineReader.read_pipelines (
    self )
```

Definition at line 20 of file producer.py.

6.9.4 Member Data Documentation

6.9.4.1 `file_path`

```
codar.savanna.producer.JSONFilePipelineReader.file_path
```

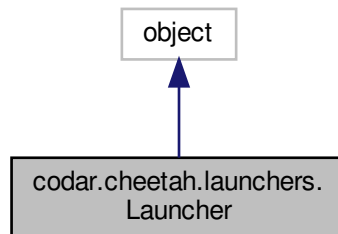
Definition at line 18 of file producer.py.

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

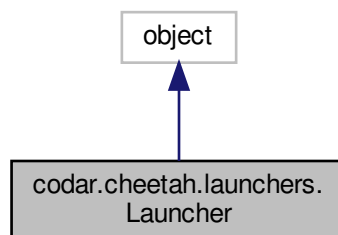
- savanna/[producer.py](#)

6.10 codar.cheetah.launchers.Launcher Class Reference

Inheritance diagram for codar.cheetah.launchers.Launcher:



Collaboration diagram for codar.cheetah.launchers.Launcher:



Public Member Functions

- `def __init__(self, machine_name, scheduler_name, runner_name, output_directory, num_codes)`
- `def create_group_directory(self, campaign_name, app_dir, group_name, runs, max_nprocs, nodes, launch_mode, component_subdirs, walltime, node_exclusive, timeout, machine, sosd_path=None, sos_analysis_path=None, tau_config=None, kill_on_partial_failure=False, run_post_process_script=None, run_post_process_stop_on_failure=False, scheduler_options=None, run_dir_setup_script=None)`
- `def read_jobid(self)`

Public Attributes

- `machine_name`
- `scheduler_name`
- `runner_name`
- `output_directory`
- `num_codes`

Static Public Attributes

- `name` = None
- string `submit_script_name` = 'submit.sh'
- string `wait_script_name` = 'wait.sh'
- string `status_script_name` = 'status.sh'
- string `submit_out_name` = 'codar.cheetah.submit-output.txt'
- string `run_command_name` = 'codar.cheetah.run-params.txt'
- string `run_json_name` = 'codar.cheetah.run-params.json'
- string `run_out_name` = 'codar.cheetah.run-output.txt'
- `batch_script_name` = None
- string `batch_walltime_name` = 'codar.cheetah.walltime.txt'
- string `jobid_file_name` = 'codar.cheetah.jobid.txt'

6.10.1 Detailed Description

Class to represent a single batch job or submission script.
It's job is to take a scheduler group and produce a script for executing all runs within the scheduler group with the indicated scheduler parameters.

The launcher may take configuration parameters to specify which scheduler/runner to use, but there is no longer an object model for schedulers and runners.

Definition at line 28 of file `launchers.py`.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 `__init__()`

```
def codar.cheetah.launchers.Launcher.__init__ (
    self,
    machine_name,
    scheduler_name,
    runner_name,
    output_directory,
    num_codes )
```

Definition at line 54 of file `launchers.py`.

6.10.3 Member Function Documentation

6.10.3.1 create_group_directory()

```
def codar.cheetah.launchers.Launcher.create_group_directory (
    self,
    campaign_name,
    app_dir,
    group_name,
    runs,
    max_nprocs,
    nodes,
    launch_mode,
    component_subdirs,
    walltime,
    node_exclusive,
    timeout,
    machine,
    sosd_path = None,
    sos_analysis_path = None,
    tau_config = None,
    kill_on_partial_failure = False,
    run_post_process_script = None,
    run_post_process_stop_on_failure = False,
    scheduler_options = None,
    run_dir_setup_script = None )
```

Copy scripts for the appropriate scheduler to group directory,
and write environment configuration. Returns required number of nodes,
which will be calculated if the passed nodes is None

Definition at line 72 of file launchers.py.

6.10.3.2 read_jobid()

```
def codar.cheetah.launchers.Launcher.read_jobid (
    self )
```

Definition at line 400 of file launchers.py.

6.10.4 Member Data Documentation

6.10.4.1 batch_script_name

```
codar.cheetah.launchers.Launcher.batch_script_name = None [static]
```

Definition at line 49 of file launchers.py.

6.10.4.2 batch_walltime_name

```
string codar.cheetah.launchers.Launcher.batch_walltime_name = 'codar.cheetah.walltime.txt'  
[static]
```

Definition at line 50 of file launchers.py.

6.10.4.3 jobid_file_name

```
string codar.cheetah.launchers.Launcher.jobid_file_name = 'codar.cheetah.jobid.txt' [static]
```

Definition at line 51 of file launchers.py.

6.10.4.4 machine_name

```
codar.cheetah.launchers.Launcher.machine_name
```

Definition at line 55 of file launchers.py.

6.10.4.5 name

```
codar.cheetah.launchers.Launcher.name = None [static]
```

Definition at line 39 of file launchers.py.

6.10.4.6 num_codes

```
codar.cheetah.launchers.Launcher.num_codes
```

Definition at line 59 of file launchers.py.

6.10.4.7 output_directory

```
codar.cheetah.launchers.Launcher.output_directory
```

Definition at line 58 of file launchers.py.

6.10.4.8 run_command_name

```
string codar.cheetah.launchers.Launcher.run_command_name = 'codar.cheetah.run-params.txt'  
[static]
```

Definition at line 46 of file launchers.py.

6.10.4.9 run_json_name

```
string codar.cheetah.launchers.Launcher.run_json_name = 'codar.cheetah.run-params.json' [static]
```

Definition at line 47 of file launchers.py.

6.10.4.10 run_out_name

```
string codar.cheetah.launchers.Launcher.run_out_name = 'codar.cheetah.run-output.txt' [static]
```

Definition at line 48 of file launchers.py.

6.10.4.11 runner_name

```
codar.cheetah.launchers.Launcher.runner_name
```

Definition at line 57 of file launchers.py.

6.10.4.12 scheduler_name

```
codar.cheetah.launchers.Launcher.scheduler_name
```

Definition at line 56 of file launchers.py.

6.10.4.13 status_script_name

```
string codar.cheetah.launchers.Launcher.status_script_name = 'status.sh' [static]
```

Definition at line 44 of file launchers.py.

6.10.4.14 submit_out_name

```
string codar.cheetah.launchers.Launcher.submit_out_name = 'codar.cheetah.submit-output.txt'  
[static]
```

Definition at line 45 of file launchers.py.

6.10.4.15 submit_script_name

```
string codar.cheetah.launchers.Launcher.submit_script_name = 'submit.sh' [static]
```

Definition at line 42 of file launchers.py.

6.10.4.16 wait_script_name

```
string codar.cheetah.launchers.Launcher.wait_script_name = 'wait.sh' [static]
```

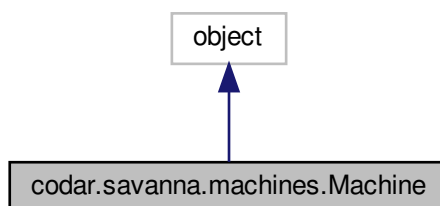
Definition at line 43 of file launchers.py.

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

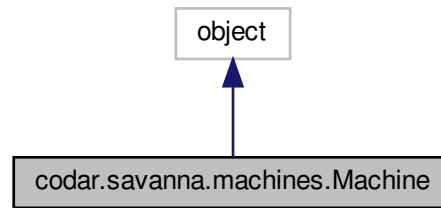
- cheetah/[launchers.py](#)

6.11 codar.savanna.machines.Machine Class Reference

Inheritance diagram for codar.savanna.machines.Machine:



Collaboration diagram for codar.savanna.machines.Machine:



Public Member Functions

- `def __init__ (self, name, scheduler_name, runner_name, node_class, processes_per_node=None, node_exclusive=False, scheduler_options=None, dataspace_servers_per_node=1)`
- `def get_scheduler_options (self, options)`
- `def get_nodes_reqd (self)`

Public Attributes

- `name`
- `scheduler_name`
- `runner_name`
- `node_class`
- `processes_per_node`
- `node_exclusive`
- `scheduler_options`
- `dataspace_servers_per_node`

6.11.1 Detailed Description

Class to represent configuration of a specific Supercomputer or workstation, including the scheduler and runner used by the machine. This can be used to map an experiment to run on the machine without having to define machine specific parameter for every experiment separately.

Definition at line 69 of file machines.py.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 `__init__()`

```
def codar.savanna.machines.Machine.__init__ (
    self,
    name,
    scheduler_name,
    runner_name,
    node_class,
    processes_per_node = None,
    node_exclusive = False,
    scheduler_options = None,
    dataspaces_servers_per_node = 1 )
```

Definition at line 78 of file machines.py.

6.11.3 Member Function Documentation

6.11.3.1 `get_nodes_reqd()`

```
def codar.savanna.machines.Machine.get_nodes_reqd (
    self )
```

Definition at line 100 of file machines.py.

6.11.3.2 `get_scheduler_options()`

```
def codar.savanna.machines.Machine.get_scheduler_options (
    self,
    options )
```

Validate supplied options and add default values where missing.
Returns a new dictionary.

Definition at line 91 of file machines.py.

6.11.4 Member Data Documentation

6.11.4.1 `dataspaces_servers_per_node`

```
codar.savanna.machines.Machine.dataspaces_servers_per_node
```

Definition at line 89 of file machines.py.

6.11.4.2 name

`codar.savanna.machines.Machine.name`

Definition at line 79 of file machines.py.

6.11.4.3 node_class

`codar.savanna.machines.Machine.node_class`

Definition at line 82 of file machines.py.

6.11.4.4 node_exclusive

`codar.savanna.machines.Machine.node_exclusive`

Definition at line 86 of file machines.py.

6.11.4.5 processes_per_node

`codar.savanna.machines.Machine.processes_per_node`

Definition at line 85 of file machines.py.

6.11.4.6 runner_name

`codar.savanna.machines.Machine.runner_name`

Definition at line 81 of file machines.py.

6.11.4.7 scheduler_name

`codar.savanna.machines.Machine.scheduler_name`

Definition at line 80 of file machines.py.

6.11.4.8 scheduler_options

`codar.savanna.machines.Machine.scheduler_options`

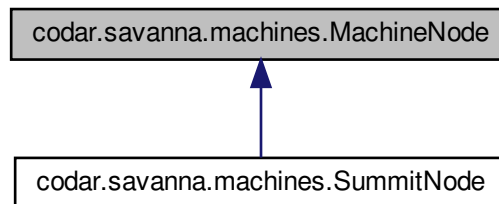
Definition at line 88 of file `machines.py`.

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

- [savanna/machines.py](#)

6.12 `codar.savanna.machines.MachineNode` Class Reference

Inheritance diagram for `codar.savanna.machines.MachineNode`:



Public Member Functions

- `def __init__(self, num_cpus, num_gpus)`
- `def validate_layout(self)`
- `def to_json(self)`

Public Attributes

- `cpu`
- `gpu`

6.12.1 Detailed Description

Definition at line 16 of file `machines.py`.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 `__init__()`

```
def codar.savanna.machines.MachineNode.__init__ (
    self,
    num_cpus,
    num_gpus )
```

Definition at line 17 of file machines.py.

6.12.3 Member Function Documentation

6.12.3.1 `to_json()`

```
def codar.savanna.machines.MachineNode.to_json (
    self )
```

Definition at line 25 of file machines.py.

6.12.3.2 `validate_layout()`

```
def codar.savanna.machines.MachineNode.validate_layout (
    self )
```

Definition at line 22 of file machines.py.

6.12.4 Member Data Documentation

6.12.4.1 `cpu`

```
codar.savanna.machines.MachineNode.cpu
```

Definition at line 19 of file machines.py.

6.12.4.2 `gpu`

```
codar.savanna.machines.MachineNode.gpu
```

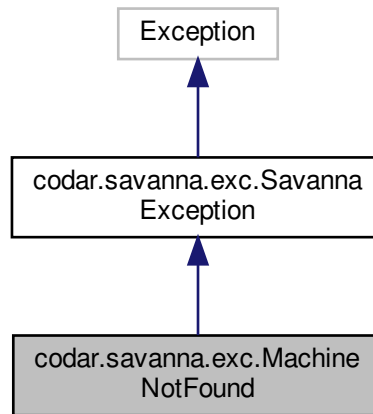
Definition at line 20 of file machines.py.

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

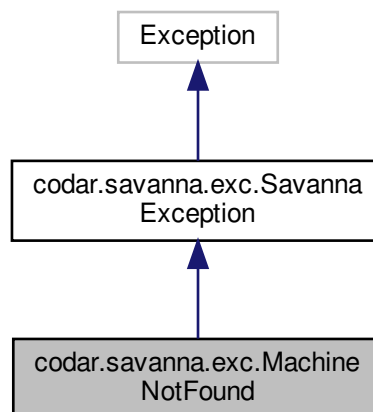
- savanna/[machines.py](#)

6.13 codar.savanna.exc.MachineNotFound Class Reference

Inheritance diagram for codar.savanna.exc.MachineNotFound:



Collaboration diagram for codar.savanna.exc.MachineNotFound:



Public Member Functions

- `def __init__(self, machine_name)`

6.13.1 Detailed Description

Definition at line 10 of file exc.py.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 `__init__()`

```
def codar.savanna.exc.MachineNotFound.__init__ (
    self,
    machine_name )
```

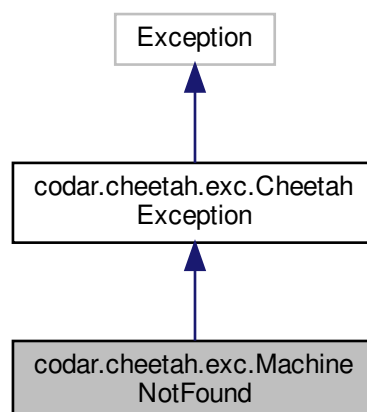
Definition at line 11 of file exc.py.

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

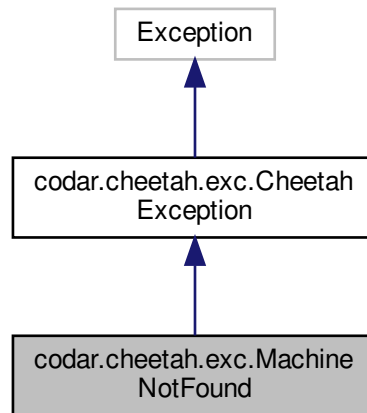
- savanna/[exc.py](#)

6.14 codar.cheetah.exc.MachineNotFound Class Reference

Inheritance diagram for codar.cheetah.exc.MachineNotFound:



Collaboration diagram for `codar.cheetah.exc.MachineNotFound`:



Public Member Functions

- `def __init__(self, machine_name)`

6.14.1 Detailed Description

Definition at line 10 of file `exc.py`.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 `__init__()`

```
def codar.cheetah.exc.MachineNotFound.__init__ (
    self,
    machine_name )
```

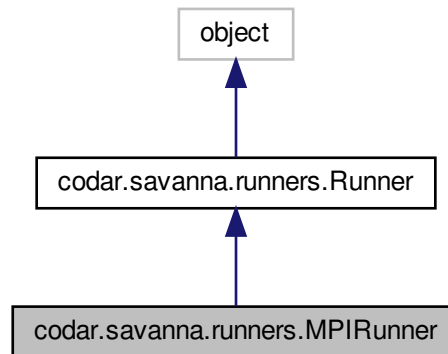
Definition at line 11 of file `exc.py`.

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

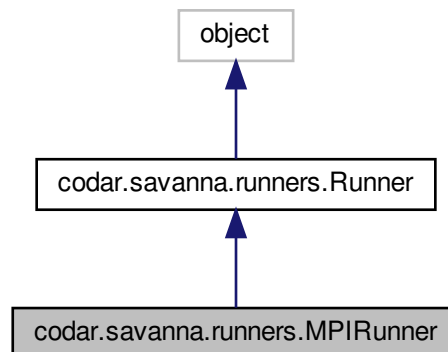
- `cheetah/exc.py`

6.15 codar.savanna.runners.MPIRunner Class Reference

Inheritance diagram for codar.savanna.runners.MPIRunner:



Collaboration diagram for codar.savanna.runners.MPIRunner:



Public Member Functions

- def `__init__` (self, `exe`, `nprocs_arg`, `nodes_arg`=None, `tasks_per_node_arg`=None, `hostfile`=None)
- def `wrap` (self, run, sched_args, find_in_path=True)

Public Attributes

- `exe`
- `nprocs_arg`
- `nodes_arg`
- `tasks_per_node_arg`
- `hostfile`

6.15.1 Detailed Description

Definition at line 11 of file runners.py.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 `__init__()`

```
def codar.savanna.runners.MPIRunner.__init__ (
    self,
    exe,
    nprocs_arg,
    nodes_arg = None,
    tasks_per_node_arg = None,
    hostfile = None )
```

Definition at line 13 of file runners.py.

6.15.3 Member Function Documentation

6.15.3.1 `wrap()`

```
def codar.savanna.runners.MPIRunner.wrap (
    self,
    run,
    sched_args,
    find_in_path = True )
```

Definition at line 20 of file runners.py.

6.15.4 Member Data Documentation

6.15.4.1 `exe`

```
codar.savanna.runners.MPIRunner.exe
```

Definition at line 14 of file runners.py.

6.15.4.2 hostfile

`codar.savanna.runners.MPIRunner.hostfile`

Definition at line 18 of file runners.py.

6.15.4.3 nodes_arg

`codar.savanna.runners.MPIRunner.nodes_arg`

Definition at line 16 of file runners.py.

6.15.4.4 nprocs_arg

`codar.savanna.runners.MPIRunner.nprocs_arg`

Definition at line 15 of file runners.py.

6.15.4.5 tasks_per_node_arg

`codar.savanna.runners.MPIRunner.tasks_per_node_arg`

Definition at line 17 of file runners.py.

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

- savanna/[runners.py](#)

6.16 codar.savanna.model.NodeConfig Class Reference

Public Member Functions

- [def __init__\(self\)](#)

Public Attributes

- [num_ranks_per_node](#)
- [cpu](#)
- [gpu](#)

6.16.1 Detailed Description

Definition at line 52 of file model.py.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 `__init__()`

```
def codar.savanna.model.NodeConfig.__init__(  
    self )
```

Intended to look like
`cpu = [0=[], 1=[], 2=[], 3=[]]`
`gpu = [0=[], 1=[], 2=[], 3=[]]`

Definition at line 53 of file model.py.

6.16.3 Member Data Documentation

6.16.3.1 `cpu`

```
codar.savanna.model.NodeConfig.cpu
```

Definition at line 60 of file model.py.

6.16.3.2 `gpu`

```
codar.savanna.model.NodeConfig.gpu
```

Definition at line 61 of file model.py.

6.16.3.3 `num_ranks_per_node`

```
codar.savanna.model.NodeConfig.num_ranks_per_node
```

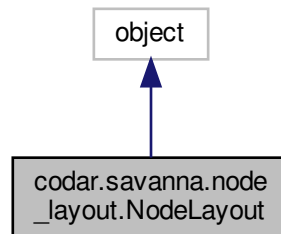
Definition at line 59 of file model.py.

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

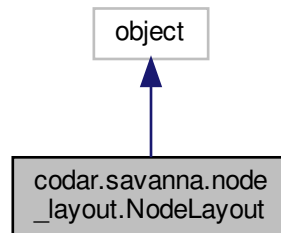
- savanna/[model.py](#)

6.17 codar.savanna.node_layout.NodeLayout Class Reference

Inheritance diagram for codar.savanna.node_layout.NodeLayout:



Collaboration diagram for codar.savanna.node_layout.NodeLayout:



Public Member Functions

- def `__init__` (self, `layout_list`)
- def `add_node` (self, `node_dict`)
- def `get_node_containing_code` (self, `code`)
- def `codes_per_node` (self)
- def `shared_nodes` (self)
- def `ppn` (self)
- def `validate` (self, `ppn`, `codes_per_node`, `shared_nodes`)
- def `as_data_list` (self)
- def `serialize_to_dict` (self)
- def `copy` (self)
- def `group_codes_by_node` (self)
- def `populate_remaining` (self, `rc_names`, `ppn`)
- def `default_no_share_layout` (cls, `ppn`, `code_names`)

Public Attributes

- [layout_list](#)
- [layout_map](#)

6.17.1 Detailed Description

Class representing options on how to organize a multi-exe task across many nodes. It is the scheduler model's job to take this and produce the correct scheduler and runner options to make this happen, or raise an error if it's not possible. Note that this will generally be different for each machine unless it is very simple and supported uniformly by all desired machines.

A layout is represented as a list of dictionaries, where each dictionary described codes to be run together on a single node. The keys are the names of the codes, and the values are the number of processes to assign to each.

Definition at line 6 of file `node_layout.py`.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 `__init__()`

```
def codar.savanna.node_layout.NodeLayout.__init__ (
    self,
    layout_list )
```

Definition at line 20 of file `node_layout.py`.

6.17.3 Member Function Documentation

6.17.3.1 `add_node()`

```
def codar.savanna.node_layout.NodeLayout.add_node (
    self,
    node_dict )
```

Add a node to an existing layout, e.g. add `sosflow`.

Definition at line 43 of file `node_layout.py`.

6.17.3.2 as_data_list()

```
def codar.savanna.node_layout.NodeLayout.as_data_list (
    self )
```

Definition at line 114 of file node_layout.py.

6.17.3.3 codes_per_node()

```
def codar.savanna.node_layout.NodeLayout.codes_per_node (
    self )
```

Definition at line 55 of file node_layout.py.

6.17.3.4 copy()

```
def codar.savanna.node_layout.NodeLayout.copy (
    self )
```

Definition at line 129 of file node_layout.py.

6.17.3.5 default_no_share_layout()

```
def codar.savanna.node_layout.NodeLayout.default_no_share_layout (
    cls,
    ppn,
    code_names )
```

Create a layout object for the specified codes and ppn, where each code uses max procs on it's own node.

Definition at line 173 of file node_layout.py.

6.17.3.6 get_node_containing_code()

```
def codar.savanna.node_layout.NodeLayout.get_node_containing_code (
    self,
    code )
```

Get node dict containing the specified code. Raises KeyError if not found.

Definition at line 50 of file node_layout.py.

6.17.3.7 group_codes_by_node()

```
def codar.savanna.node_layout.NodeLayout.group_codes_by_node (
    self )
```

Return a list of dicts, where each list represents codes on a node, and a dict key for ppn
Example: [{sim,analysis1}, {analysis2}, {viz}].
Must take Summit NodeConfigs into account

Definition at line 132 of file node_layout.py.

6.17.3.8 populate_remaining()

```
def codar.savanna.node_layout.NodeLayout.populate_remaining (
    self,
    rc_names,
    ppn )
```

Definition at line 161 of file node_layout.py.

6.17.3.9 ppn()

```
def codar.savanna.node_layout.NodeLayout.ppn (
    self )
```

Definition at line 61 of file node_layout.py.

6.17.3.10 serialize_to_dict()

```
def codar.savanna.node_layout.NodeLayout.serialize_to_dict (
    self )
```

Get a copy of the data list passed to the constructor,
suitable for JSON serialization.

Definition at line 117 of file node_layout.py.

6.17.3.11 shared_nodes()

```
def codar.savanna.node_layout.NodeLayout.shared_nodes (
    self )
```

Definition at line 58 of file node_layout.py.

6.17.3.12 validate()

```
def codar.savanna.node_layout.NodeLayout.validate (
    self,
    ppn,
    codes_per_node,
    shared_nodes )
```

Given a machine ppn and max numer of codes (e.g. 4 on cori),
raise a ValueError if the specified layout won't fit.
Dont modify this yet, this is being used by the tests

Definition at line 96 of file node_layout.py.

6.17.4 Member Data Documentation

6.17.4.1 layout_list

```
codar.savanna.node_layout.NodeLayout.layout_list
```

Definition at line 34 of file node_layout.py.

6.17.4.2 layout_map

```
codar.savanna.node_layout.NodeLayout.layout_map
```

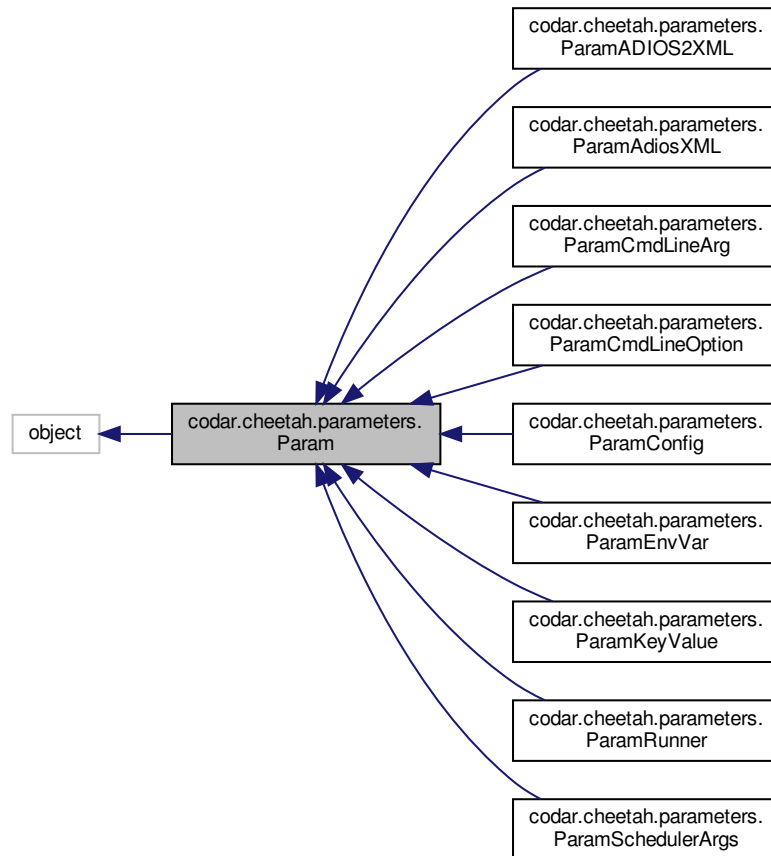
Definition at line 35 of file node_layout.py.

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

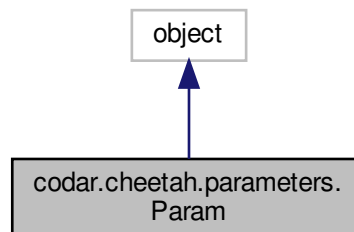
- savanna/[node_layout.py](#)

6.18 codar.cheetah.parameters.Param Class Reference

Inheritance diagram for codar.cheetah.parameters.Param:



Collaboration diagram for codar.cheetah.parameters.Param:



Public Member Functions

- `def __init__ (self, target, name, values)`
- `def __get__ (self, idx)`
- `def __len__ (self)`

Public Attributes

- `target`
- `name`
- `values`

6.18.1 Detailed Description

Abstract base class representing a parameter to an application. This includes any method for modifying the run characteristics of an application - command line, config file, environment variables, different executable built with different compiler flags.

Every parameter must have a unique name, and must target a specific application or middleware, e.g. pbs, aprun, or one of the science codes that make up an application.

Note that if a science application has only one code, it will likely still involve middleware targets like PBS. Using a different target is one way to model those.

TODO: is it useful to separate the definition of a param and it's values?

TODO: should we require that the name be unique across all targets, or just within each target? Global uniqueness allows for a simple list of dict representation of instances, but two level nested dicts may be more powerful (first level is target, second level is params).

Definition at line 299 of file parameters.py.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 __init__()

```
def codar.cheetah.parameters.Param.__init__ (
    self,
    target,
    name,
    values )
```

Definition at line 320 of file parameters.py.

6.18.3 Member Function Documentation

6.18.3.1 `__get__()`

```
def codar.cheetah.parameters.Param.__get__ (
    self,
    idx )
```

Definition at line 334 of file parameters.py.

6.18.3.2 `__len__()`

```
def codar.cheetah.parameters.Param.__len__ (
    self )
```

Definition at line 337 of file parameters.py.

6.18.4 Member Data Documentation

6.18.4.1 `name`

```
codar.cheetah.parameters.Param.name
```

Definition at line 322 of file parameters.py.

6.18.4.2 `target`

```
codar.cheetah.parameters.Param.target
```

Definition at line 321 of file parameters.py.

6.18.4.3 `values`

```
codar.cheetah.parameters.Param.values
```

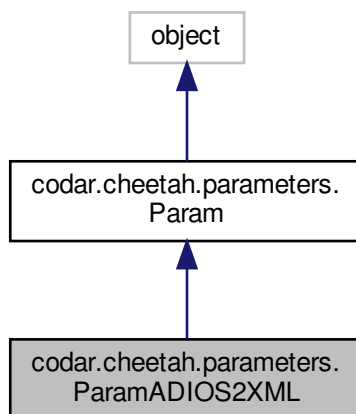
Definition at line 331 of file parameters.py.

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

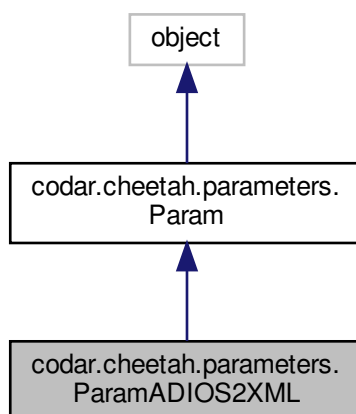
- cheetah/[parameters.py](#)

6.19 codar.cheetah.parameters.ParamADIOS2XML Class Reference

Inheritance diagram for codar.cheetah.parameters.ParamADIOS2XML:



Collaboration diagram for codar.cheetah.parameters.ParamADIOS2XML:



Public Member Functions

- `def __init__(self, rc, io_name, operation_name, values)`

Public Attributes

- [rc](#)
- [io_name](#)
- [operation_name](#)
- [values](#)

6.19.1 Detailed Description

Class to represent ADIOS2 XML file parameter options

Definition at line 376 of file parameters.py.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamADIOS2XML.__init__ (
    self,
    rc,
    io_name,
    operation_name,
    values )
```

```
:param rc: name of the run component
:param io_name: name of the io object in the xml file
:param operation_name: engine/transport/var_operation
:param values: a list of dicts of the type
[ { engine_name: {parameters} },
  { engine_name: {parameters} },
  { var_name: {operation_name: {parameters}}} ]
```

Examples:

```
[ { "BPFile": { 'Threads': 1 } },
  { "BPFile": { "ProfileUnits": "Microseconds" } } ]
[ { "T": { "zfp": { "rate": 18, "accuracy": 0.01 } } },
  { "T": { "zfp": { "rate": 18, "accuracy": 0.001 } } },
  { "T": { "zfp": { "rate": 18, "accuracy": 0.0001 } } },
  { "T": { "sz": { "rate": 18, "accuracy": 0.01 } } },
]
```

Definition at line 380 of file parameters.py.

6.19.3 Member Data Documentation

6.19.3.1 io_name

`codar.cheetah.parameters.ParamADIOS2XML.io_name`

Definition at line 404 of file parameters.py.

6.19.3.2 operation_name

`codar.cheetah.parameters.ParamADIOS2XML.operation_name`

Definition at line 405 of file parameters.py.

6.19.3.3 rc

`codar.cheetah.parameters.ParamADIOS2XML.rc`

Definition at line 403 of file parameters.py.

6.19.3.4 values

`codar.cheetah.parameters.ParamADIOS2XML.values`

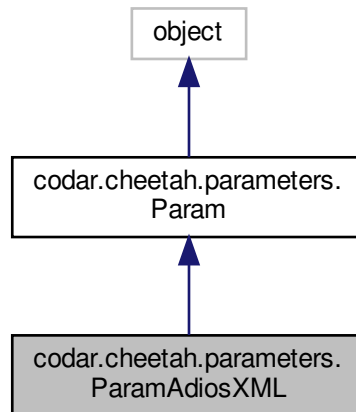
Definition at line 406 of file parameters.py.

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

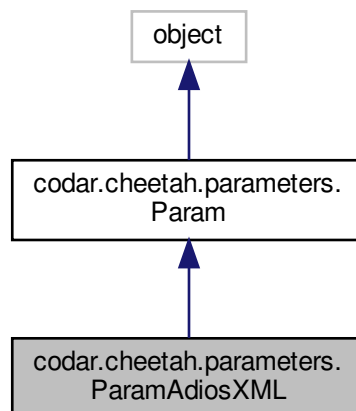
- cheetah/[parameters.py](#)

6.20 `codar.cheetah.parameters.ParamAdiosXML` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamAdiosXML`:



Collaboration diagram for `codar.cheetah.parameters.ParamAdiosXML`:



Public Member Functions

- `def __init__(self, target, name, adios_xml_tags, values)`

Public Attributes

- [param_type](#)
- [group_name](#)
- [var_name](#)

6.20.1 Detailed Description

Class to represent ADIOS XML Transform.

The transform config is encoded in the name, so transforms on different variables can be included in the sweep.

Format:

```
adios_transform:<group_name>:<var_name>
adios_transport:<group_name>
```

Note that the filename is specified in the code definition.

Definition at line 349 of file parameters.py.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamAdiosXML.__init__ (
    self,
    target,
    name,
    adios_xml_tags,
    values )
```

Definition at line 362 of file parameters.py.

6.20.3 Member Data Documentation

6.20.3.1 `group_name`

`codar.cheetah.parameters.ParamAdiosXML.group_name`

Definition at line 369 of file parameters.py.

6.20.3.2 param_type

`codar.cheetah.parameters.ParamAdiosXML.param_type`

Definition at line 368 of file parameters.py.

6.20.3.3 var_name

`codar.cheetah.parameters.ParamAdiosXML.var_name`

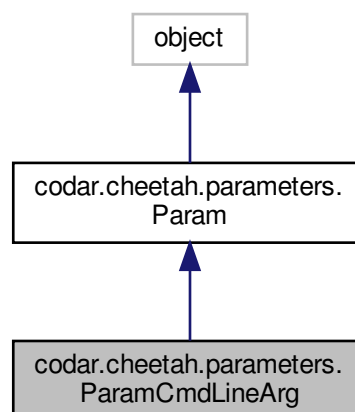
Definition at line 373 of file parameters.py.

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

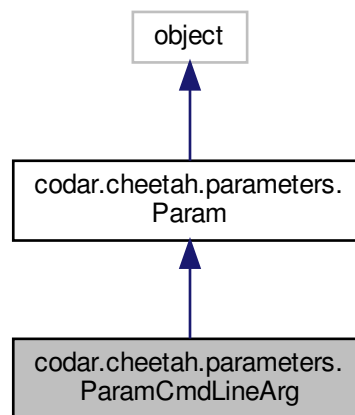
- [cheetah/parameters.py](#)

6.21 codar.cheetah.parameters.ParamCmdLineArg Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamCmdLineArg`:



Collaboration diagram for codar.cheetah.parameters.ParamCmdLineArg:



Public Member Functions

- `def __init__(self, target, name, position, values)`

Public Attributes

- `position`

6.21.1 Detailed Description

Specification for parameters that are based as a positional command line argument.

Definition at line 341 of file parameters.py.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 __init__()

```
def codar.cheetah.parameters.ParamCmdLineArg.__init__(  
    self,  
    target,  
    name,  
    position,  
    values )
```

Definition at line 344 of file parameters.py.

6.21.3 Member Data Documentation

6.21.3.1 position

`codar.cheetah.parameters.ParamCmdLineArg.position`

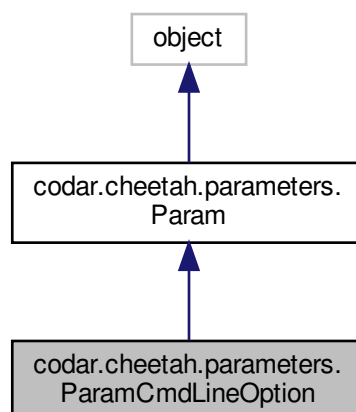
Definition at line 346 of file `parameters.py`.

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

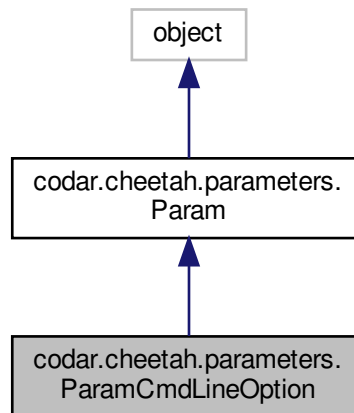
- [cheetah/parameters.py](#)

6.22 `codar.cheetah.parameters.ParamCmdLineOption` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamCmdLineOption`:



Collaboration diagram for codar.cheetah.parameters.ParamCmdLineOption:



Public Member Functions

- `def __init__(self, target, name, option, values)`

Public Attributes

- `option`

6.22.1 Detailed Description

Specification for parameters that are based as a labeled command line option. The option must contain the prefix, e.g. `'--output-file'` not `'output-file'`.

Definition at line 455 of file parameters.py.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamCmdLineOption.__init__(
    self,
    target,
    name,
    option,
    values )
```

Definition at line 460 of file parameters.py.

6.22.3 Member Data Documentation

6.22.3.1 option

`codar.cheetah.parameters.ParamCmdLineOption.option`

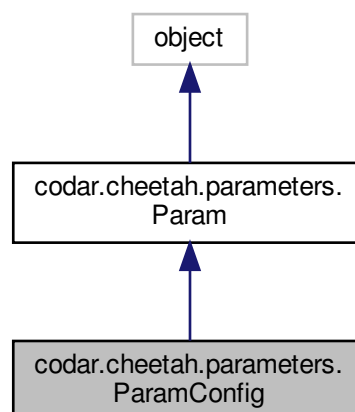
Definition at line 462 of file `parameters.py`.

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

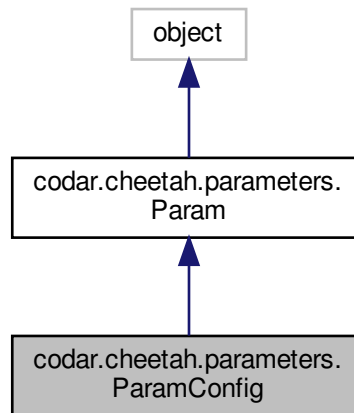
- [cheetah/parameters.py](#)

6.23 `codar.cheetah.parameters.ParamConfig` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamConfig`:



Collaboration diagram for codar.cheetah.parameters.ParamConfig:



Public Member Functions

- `def __init__(self, target, name, config_filename, match_string, values)`

Public Attributes

- `config_filename`
- `match_string`

6.23.1 Detailed Description

Class to represent a simple literal string replace in a config file.

Note that the filename must be added to the inputs list as well, to be copied to each run directory.

Definition at line 423 of file parameters.py.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamConfig.__init__ (
    self,
    target,
    name,
    config_filename,
    match_string,
    values )
```

Definition at line 430 of file parameters.py.

6.23.3 Member Data Documentation

6.23.3.1 `config_filename`

```
codar.cheetah.parameters.ParamConfig.config_filename
```

Definition at line 432 of file parameters.py.

6.23.3.2 `match_string`

```
codar.cheetah.parameters.ParamConfig.match_string
```

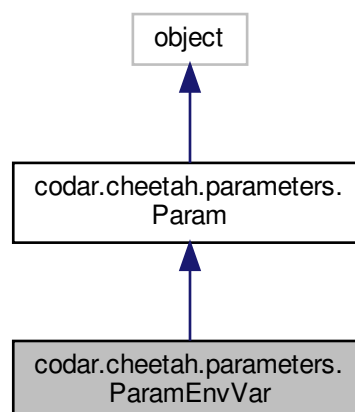
Definition at line 433 of file parameters.py.

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

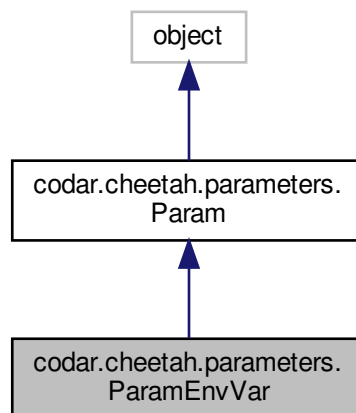
- cheetah/[parameters.py](#)

6.24 `codar.cheetah.parameters.ParamEnvVar` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamEnvVar`:



Collaboration diagram for codar.cheetah.parameters.ParamEnvVar:



Public Member Functions

- `def __init__ (self, target, name, option, values)`

Public Attributes

- [option](#)

6.24.1 Detailed Description

Definition at line 465 of file `parameters.py`.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamEnvVar.__init__ (
    self,
    target,
    name,
    option,
    values )
```

Definition at line 466 of file `parameters.py`.

6.24.3 Member Data Documentation

6.24.3.1 option

`codar.cheetah.parameters.ParamEnvVar.option`

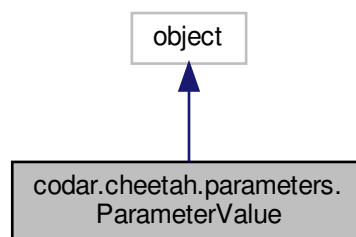
Definition at line 468 of file `parameters.py`.

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

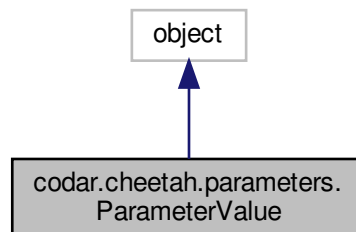
- [cheetah/parameters.py](#)

6.25 `codar.cheetah.parameters.ParameterValue` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParameterValue`:



Collaboration diagram for `codar.cheetah.parameters.ParameterValue`:



Public Member Functions

- def `__init__` (self, parameter, value_index)
- def `__getattr__` (self, name)
- def `is_type` (self, parameter_class)

Public Attributes

- `value`

6.25.1 Detailed Description

Convenience classes for tracking a specific value of a parameter. Proxies to underlying parameter object, adds a 'value' instance variable.

TODO: this is kind of hacky, is there a better way?

Definition at line 80 of file parameters.py.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 `__init__()`

```
def codar.cheetah.parameters.ParameterValue.__init__ (
    self,
    parameter,
    value_index )
```

Definition at line 88 of file parameters.py.

6.25.3 Member Function Documentation

6.25.3.1 `__getattr__()`

```
def codar.cheetah.parameters.ParameterValue.__getattr__ (
    self,
    name )
```

Definition at line 92 of file parameters.py.

6.25.3.2 `is_type()`

```
def codar.cheetah.parameters.ParameterValue.is_type (
    self,
    parameter_class )
```

Definition at line 97 of file parameters.py.

6.25.4 Member Data Documentation

6.25.4.1 `value`

```
codar.cheetah.parameters.ParameterValue.value
```

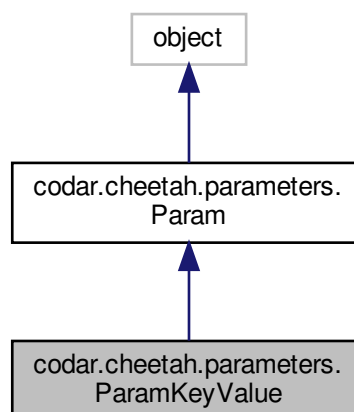
Definition at line 90 of file parameters.py.

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

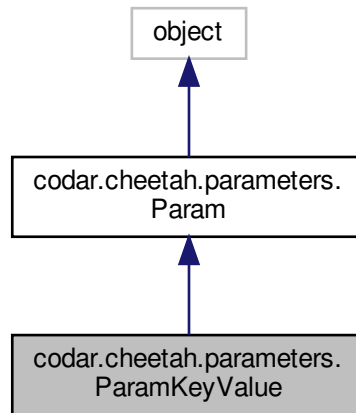
- cheetah/[parameters.py](#)

6.26 `codar.cheetah.parameters.ParamKeyValue` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamKeyValue`:



Collaboration diagram for codar.cheetah.parameters.ParamKeyValue:



Public Member Functions

- `def __init__(self, target, name, config_filename, key_name, values)`

Public Attributes

- `config_filename`
- `key_name`

6.26.1 Detailed Description

Class to represent replacement of the value in a config file with 'k = v' formatted lines. This should work with various formats, including fortran namelist and INI, by ignoring lines that don't match the simple k = v pattern. It has the advantage of being flexible, but the disadvantage of not understanding sections or other more complicated structure in config files. Also does not do any quoting - if required, the spec writer should include literal quotes around the values.

Note that the filename must be added to the inputs list as well, to be copied to each run directory.

Definition at line 436 of file parameters.py.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamKeyValue.__init__ (
    self,
    target,
    name,
    config_filename,
    key_name,
    values )
```

Definition at line 449 of file parameters.py.

6.26.3 Member Data Documentation

6.26.3.1 `config_filename`

```
codar.cheetah.parameters.ParamKeyValue.config_filename
```

Definition at line 451 of file parameters.py.

6.26.3.2 `key_name`

```
codar.cheetah.parameters.ParamKeyValue.key_name
```

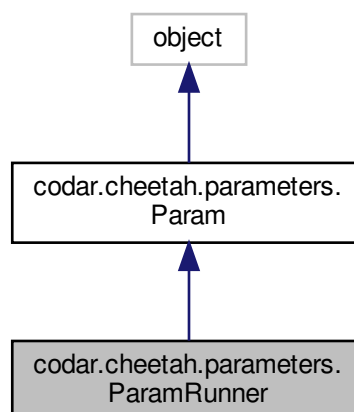
Definition at line 452 of file parameters.py.

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

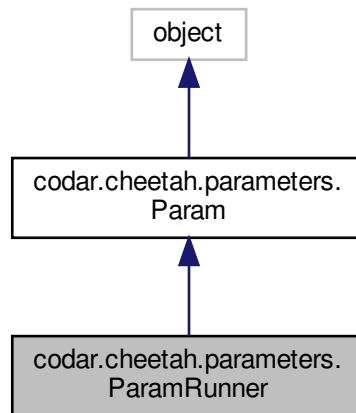
- cheetah/[parameters.py](#)

6.27 `codar.cheetah.parameters.ParamRunner` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamRunner`:



Collaboration diagram for codar.cheetah.parameters.ParamRunner:



Public Member Functions

- `def __init__(self, target, name, values)`

Additional Inherited Members

6.27.1 Detailed Description

Specification for parameters that are passed to the runner, e.g. `mpirun`, `mpilaunch`, `srun`, `apirun`, but usually still associated with a specific application code.

Definition at line 478 of file `parameters.py`.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamRunner.__init__(
    self,
    target,
    name,
    values )
```

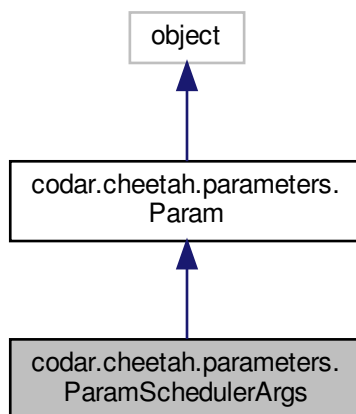
Definition at line 482 of file `parameters.py`.

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

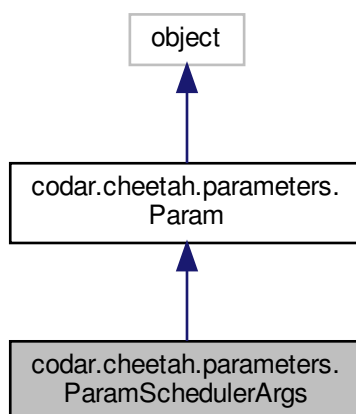
- `cheetah/parameters.py`

6.28 `codar.cheetah.parameters.ParamSchedulerArgs` Class Reference

Inheritance diagram for `codar.cheetah.parameters.ParamSchedulerArgs`:



Collaboration diagram for `codar.cheetah.parameters.ParamSchedulerArgs`:



Public Member Functions

- `def __init__(self, target, values)`

Additional Inherited Members

6.28.1 Detailed Description

Definition at line 471 of file parameters.py.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 `__init__()`

```
def codar.cheetah.parameters.ParamSchedulerArgs.__init__ (
    self,
    target,
    values )
```

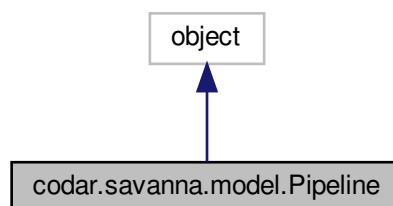
Definition at line 472 of file parameters.py.

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

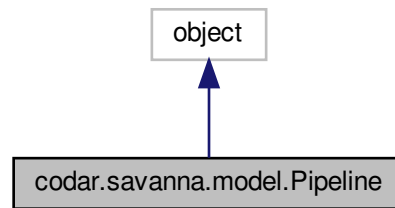
- cheetah/[parameters.py](#)

6.29 codar.savanna.model.Pipeline Class Reference

Inheritance diagram for codar.savanna.model.Pipeline:



Collaboration diagram for `codar.savanna.model.Pipeline`:



Public Member Functions

- `def __init__ (self, pipe_id, runs, working_dir, total_nodes, machine_name, kill_on_partial_failure=False, post_process_script=None, post_process_args=None, post_process_stop_on_failure=False, node_layout=None, launch_mode=None)`
- `def from_data (cls, data)`
- `def start (self, consumer, nodes_assigned, runner=None)`
- `def run_finished (self, run)`
- `def run_post_process_script (self)`
- `def add_done_callback (self, fn)`
- `def remove_done_callback (self, fn)`
- `def add_fatal_callback (self, fn)`
- `def remove_fatal_callback (self, fn)`
- `def get_nodes_used (self)`
- `def set_ppn (self, ppn)`
- `def set_total_nodes (self)`
- `def get_state (self)`
- `def get_pids (self)`
- `def force_kill_all (self)`
- `def join_all (self)`

Public Attributes

- `id`
- `runs`
- `working_dir`
- `kill_on_partial_failure`
- `post_process_script`
- `post_process_args`
- `post_process_stop_on_failure`
- `node_layout`
- `machine_name`
- `done_callbacks`
- `fatal_callbacks`
- `total_procs`
- `log_prefix`
- `total_nodes`
- `launch_mode`
- `nodes_assigned`

6.29.1 Detailed Description

Definition at line 449 of file model.py.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 __init__()

```
def codar.savanna.model.Pipeline.__init__ (
    self,
    pipe_id,
    runs,
    working_dir,
    total_nodes,
    machine_name,
    kill_on_partial_failure = False,
    post_process_script = None,
    post_process_args = None,
    post_process_stop_on_failure = False,
    node_layout = None,
    launch_mode = None )
```

Definition at line 455 of file model.py.

6.29.3 Member Function Documentation

6.29.3.1 add_done_callback()

```
def codar.savanna.model.Pipeline.add_done_callback (
    self,
    fn )
```

Definition at line 818 of file model.py.

6.29.3.2 add_fatal_callback()

```
def codar.savanna.model.Pipeline.add_fatal_callback (
    self,
    fn )
```

Definition at line 830 of file model.py.

6.29.3.3 force_kill_all()

```
def codar.savanna.model.Pipeline.force_kill_all (
    self )
```

Kill all runs and don't run post processing. Note that this call may block waiting for all runs to be started, to avoid confusing races. If the pipeline is already done, this does nothing. If one or more runs are still active, or have not yet been marked as finished, then it will mark the entire pipeline as killed so it can be re-run from scratch on a restart if desired.

Definition at line 912 of file model.py.

6.29.3.4 from_data()

```
def codar.savanna.model.Pipeline.from_data (
    cls,
    data )
```

Create Pipeline instance from dictionary data structure, containing at least "id" and "runs" keys. The "runs" key must have a list of dict, and each dict is parsed using Run.from_data. Raises KeyError if a required key is missing.

Definition at line 497 of file model.py.

6.29.3.5 get_nodes_used()

```
def codar.savanna.model.Pipeline.get_nodes_used (
    self )
```

Definition at line 850 of file model.py.

6.29.3.6 get_pids()

```
def codar.savanna.model.Pipeline.get_pids (
    self )
```

Definition at line 908 of file model.py.

6.29.3.7 get_state()

```
def codar.savanna.model.Pipeline.get_state (
    self )
```

Definition at line 883 of file model.py.

6.29.3.8 join_all()

```
def codar.savanna.model.Pipeline.join_all (
    self )
```

Definition at line 933 of file model.py.

6.29.3.9 remove_done_callback()

```
def codar.savanna.model.Pipeline.remove_done_callback (
    self,
    fn )
```

Definition at line 821 of file model.py.

6.29.3.10 remove_fatal_callback()

```
def codar.savanna.model.Pipeline.remove_fatal_callback (
    self,
    fn )
```

Definition at line 833 of file model.py.

6.29.3.11 run_finished()

```
def codar.savanna.model.Pipeline.run_finished (
    self,
    run )
```

Definition at line 744 of file model.py.

6.29.3.12 run_post_process_script()

```
def codar.savanna.model.Pipeline.run_post_process_script (
    self )
```

Definition at line 772 of file model.py.

6.29.3.13 set_ppn()

```
def codar.savanna.model.Pipeline.set_ppn (
    self,
    ppn )
```

Determine number of nodes needed to run pipeline with the specified node layout or full occupancy layout with ppn. Also updates runs to set node and task per node counts.
TODO: This should be set by Cheetah in fobs.json

Definition at line 855 of file model.py.

6.29.3.14 set_total_nodes()

```
def codar.savanna.model.Pipeline.set_total_nodes (
    self )
```

To be deprecated

Definition at line 877 of file model.py.

6.29.3.15 start()

```
def codar.savanna.model.Pipeline.start (
    self,
    consumer,
    nodes_assigned,
    runner = None )
```

Definition at line 550 of file model.py.

6.29.4 Member Data Documentation

6.29.4.1 done_callbacks

`codar.savanna.model.Pipeline.done_callbacks`

Definition at line 473 of file model.py.

6.29.4.2 fatal_callbacks

`codar.savanna.model.Pipeline.fatal_callbacks`

Definition at line 474 of file model.py.

6.29.4.3 id

`codar.savanna.model.Pipeline.id`

Definition at line 456 of file model.py.

6.29.4.4 kill_on_partial_failure

`codar.savanna.model.Pipeline.kill_on_partial_failure`

Definition at line 459 of file model.py.

6.29.4.5 launch_mode

`codar.savanna.model.Pipeline.launch_mode`

Definition at line 483 of file model.py.

6.29.4.6 log_prefix

`codar.savanna.model.Pipeline.log_prefix`

Definition at line 476 of file model.py.

6.29.4.7 machine_name

`codar.savanna.model.Pipeline.machine_name`

Definition at line 464 of file model.py.

6.29.4.8 node_layout

`codar.savanna.model.Pipeline.node_layout`

Definition at line 463 of file model.py.

6.29.4.9 nodes_assigned

`codar.savanna.model.Pipeline.nodes_assigned`

Definition at line 487 of file model.py.

6.29.4.10 post_process_args

`codar.savanna.model.Pipeline.post_process_args`

Definition at line 461 of file model.py.

6.29.4.11 post_process_script

`codar.savanna.model.Pipeline.post_process_script`

Definition at line 460 of file model.py.

6.29.4.12 post_process_stop_on_failure

`codar.savanna.model.Pipeline.post_process_stop_on_failure`

Definition at line 462 of file model.py.

6.29.4.13 runs

`codar.savanna.model.Pipeline.runs`

Definition at line 457 of file `model.py`.

6.29.4.14 total_nodes

`codar.savanna.model.Pipeline.total_nodes`

Definition at line 482 of file `model.py`.

6.29.4.15 total_procs

`codar.savanna.model.Pipeline.total_procs`

Definition at line 475 of file `model.py`.

6.29.4.16 working_dir

`codar.savanna.model.Pipeline.working_dir`

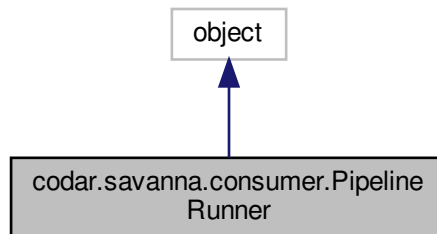
Definition at line 458 of file `model.py`.

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

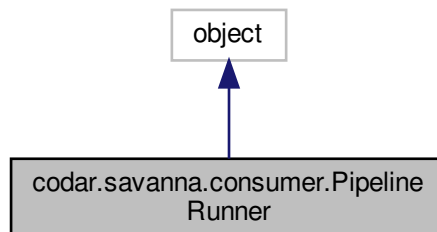
- `savanna/model.py`

6.30 `codar.savanna.consumer.PipelineRunner` Class Reference

Inheritance diagram for `codar.savanna.consumer.PipelineRunner`:



Collaboration diagram for `codar.savanna.consumer.PipelineRunner`:



Public Member Functions

- `def __init__ (self, runner, max_nodes, machine_name, processes_per_node, status_file=None)`
- `def add_pipeline (self, p)`
- `def stop (self)`
- `def kill_all (self)`
- `def run_finished (self, run)`
- `def pipeline_finished (self, pipeline)`
- `def pipeline_fatal (self, pipeline)`
- `def run_pipelines (self)`

Public Attributes

- [max_nodes](#)
- [machine_name](#)
- [ppn](#)
- [runner](#)
- [job_list_cv](#)
- [job_list](#)
- [free_cv](#)
- [free_nodes](#)
- [pipelines_lock](#)
- [pipelines](#)
- [allocated_nodes](#)

6.30.1 Detailed Description

Runner that assumes a homogenous set of nodes. Now only support only node based limiting (although process limiting can be emulated by setting `process_per_node=1` and `max_nodes=max_procs`).

Threading model: assumes there could be multiple producer threads calling `add_pipeline`, e.g. if using a dynamic job submission model based on results of previous jobs. Pipelines and each Run in a pipeline are all executed in separate threads, so their notification callbacks execute in separate threads, and their threads must be joined before exiting. The `stop` and `kill_all` methods could be called from any of the producer, Pipeline or Run threads.

Definition at line 18 of file `consumer.py`.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 `__init__()`

```
def codar.savanna.consumer.PipelineRunner.__init__ (
    self,
    runner,
    max_nodes,
    machine_name,
    processes_per_node,
    status_file = None )
```

Definition at line 32 of file `consumer.py`.

6.30.3 Member Function Documentation

6.30.3.1 add_pipeline()

```
def codar.savanna.consumer.PipelineRunner.add_pipeline (
    self,
    p )
```

Definition at line 73 of file consumer.py.

6.30.3.2 kill_all()

```
def codar.savanna.consumer.PipelineRunner.kill_all (
    self )
```

Kill all running processes spawned by this consumer and don't start any new processes.

Definition at line 114 of file consumer.py.

6.30.3.3 pipeline_fatal()

```
def codar.savanna.consumer.PipelineRunner.pipeline_fatal (
    self,
    pipeline )
```

Definition at line 190 of file consumer.py.

6.30.3.4 pipeline_finished()

```
def codar.savanna.consumer.PipelineRunner.pipeline_finished (
    self,
    pipeline )
```

Monitor thread(s) should call this as pipelines complete.

Definition at line 164 of file consumer.py.

6.30.3.5 run_finished()

```
def codar.savanna.consumer.PipelineRunner.run_finished (
    self,
    run )
```

TO BE DEPRECATED.

Monitor thread(s) should call this as runs complete. To be deprecated, as the functionality fails when node_layout is set to node-sharing.

This means that for node_exclusive, resources held by a run are not released when the run terminates. For kill_on_partial_failure=False, this could lead to unused resources, which is ok.

Definition at line 148 of file consumer.py.

6.30.3.6 run_pipelines()

```
def codar.savanna.consumer.PipelineRunner.run_pipelines (
    self )
```

Main loop of consumer thread. Does not return until all child threads are complete.

Definition at line 194 of file consumer.py.

6.30.3.7 stop()

```
def codar.savanna.consumer.PipelineRunner.stop (
    self )
```

Signal to stop when all pipelines are finished. Don't allow adding new pipelines.

Definition at line 105 of file consumer.py.

6.30.4 Member Data Documentation

6.30.4.1 allocated_nodes

```
codar.savanna.consumer.PipelineRunner.allocated_nodes
```

Definition at line 60 of file consumer.py.

6.30.4.2 free_cv

`codar.savanna.consumer.PipelineRunner.free_cv`

Definition at line 47 of file consumer.py.

6.30.4.3 free_nodes

`codar.savanna.consumer.PipelineRunner.free_nodes`

Definition at line 48 of file consumer.py.

6.30.4.4 job_list

`codar.savanna.consumer.PipelineRunner.job_list`

Definition at line 45 of file consumer.py.

6.30.4.5 job_list_cv

`codar.savanna.consumer.PipelineRunner.job_list_cv`

Definition at line 43 of file consumer.py.

6.30.4.6 machine_name

`codar.savanna.consumer.PipelineRunner.machine_name`

Definition at line 34 of file consumer.py.

6.30.4.7 max_nodes

`codar.savanna.consumer.PipelineRunner.max_nodes`

Definition at line 33 of file consumer.py.

6.30.4.8 pipelines

`codar.savanna.consumer.PipelineRunner.pipelines`

Definition at line 51 of file `consumer.py`.

6.30.4.9 pipelines_lock

`codar.savanna.consumer.PipelineRunner.pipelines_lock`

Definition at line 50 of file `consumer.py`.

6.30.4.10 ppn

`codar.savanna.consumer.PipelineRunner.ppn`

Definition at line 35 of file `consumer.py`.

6.30.4.11 runner

`codar.savanna.consumer.PipelineRunner.runner`

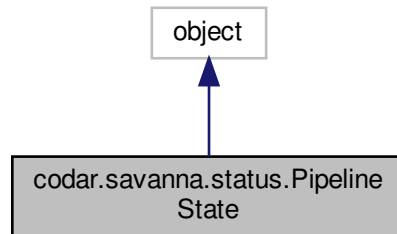
Definition at line 36 of file `consumer.py`.

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

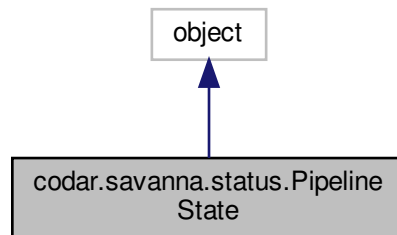
- [savanna/consumer.py](#)

6.31 `codar.savanna.status.PipelineState` Class Reference

Inheritance diagram for `codar.savanna.status.PipelineState`:



Collaboration diagram for `codar.savanna.status.PipelineState`:



Public Member Functions

- `def __init__ (self, pipeline_id, state, reason=None, return_codes=None)`
- `def as_data (self)`

Public Attributes

- `id`
- `state`
- `reason`
- `return_codes`

6.31.1 Detailed Description

Definition at line 48 of file `status.py`.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 `__init__()`

```
def codar.savanna.status.PipelineState.__init__ (
    self,
    pipeline_id,
    state,
    reason = None,
    return_codes = None )
```

Definition at line 49 of file status.py.

6.31.3 Member Function Documentation

6.31.3.1 `as_data()`

```
def codar.savanna.status.PipelineState.as_data (
    self )
```

Definition at line 55 of file status.py.

6.31.4 Member Data Documentation

6.31.4.1 `id`

```
codar.savanna.status.PipelineState.id
```

Definition at line 50 of file status.py.

6.31.4.2 `reason`

```
codar.savanna.status.PipelineState.reason
```

Definition at line 52 of file status.py.

6.31.4.3 return_codes

`codar.savanna.status.PipelineState.return_codes`

Definition at line 53 of file `status.py`.

6.31.4.4 state

`codar.savanna.status.PipelineState.state`

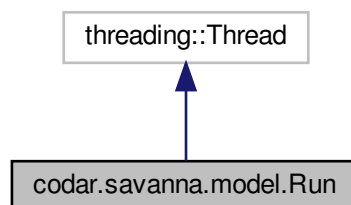
Definition at line 51 of file `status.py`.

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

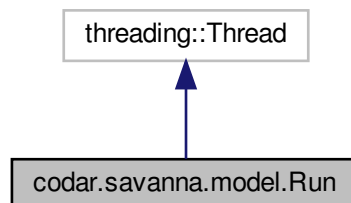
- [savanna/status.py](#)

6.32 codar.savanna.model.Run Class Reference

Inheritance diagram for `codar.savanna.model.Run`:



Collaboration diagram for `codar.savanna.model.Run`:



Public Member Functions

- def `__init__` (self, `name`, `exe`, `args`, `sched_args`, `env`, `working_dir`, `timeout`=None, `nprocs`=1, `res_set`=None, `stdout_path`=None, `stderr_path`=None, `return_path`=None, `walltime_path`=None, `log_prefix`=None, `sleep_after`=None, `depends_on_runs`=None, `hostfile`=None, `runner_override`=False)
- def `from_data` (cls, data)
- def `mpmd_run` (cls, runs)
- def `set_runner` (self, `runner`)
- def `timed_out` (self)
- def `killed` (self)
- def `exception` (self)
- def `succeeded` (self)
- def `add_callback` (self, fn)
- def `remove_callback` (self, fn)
- def `run` (self)
- def `kill` (self)
- def `get_returncode` (self)
- def `get_pid` (self)
- def `close` (self)
- def `join` (self)
- def `get_nodes_used` (self)
- def `create_node_config` (self)

Public Attributes

- `name`
- `exe`
- `args`
- `sched_args`
- `env`
- `working_dir`
- `timeout`
- `nprocs`
- `res_set`
- `stdout_path`
- `stderr_path`
- `return_path`
- `walltime_path`
- `sleep_after`
- `log_prefix`
- `runner`
- `callbacks`
- `nodes`
- `tasks_per_node`
- `depends_on_runs`
- `hostfile`
- `machine`
- `nodes_assigned`
- `node_config`
- `erf_file`
- `runner_override`

6.32.1 Detailed Description

Manage running a single executable within a pipeline. When start is called, it will launch the process with Popen and call wait in the new thread with a timeout, killing if the process does not finish in time.

Definition at line 64 of file model.py.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 `__init__()`

```
def codar.savanna.model.Run.__init__ (
    self,
    name,
    exe,
    args,
    sched_args,
    env,
    working_dir,
    timeout = None,
    nprocs = 1,
    res_set = None,
    stdout_path = None,
    stderr_path = None,
    return_path = None,
    walltime_path = None,
    log_prefix = None,
    sleep_after = None,
    depends_on_runs = None,
    hostfile = None,
    runner_override = False )
```

Definition at line 74 of file model.py.

6.32.3 Member Function Documentation

6.32.3.1 `add_callback()`

```
def codar.savanna.model.Run.add_callback (
    self,
    fn )
```

Function takes single argument which is this run instance, and is called when the process is complete (either normally or killed by timeout). Callbacks must not block.

Definition at line 228 of file model.py.

6.32.3.2 close()

```
def codar.savanna.model.Run.close (
    self )
```

Definition at line 426 of file model.py.

6.32.3.3 create_node_config()

```
def codar.savanna.model.Run.create_node_config (
    self )
```

Definition at line 445 of file model.py.

6.32.3.4 exception()

```
def codar.savanna.model.Run.exception (
    self )
```

True if there was a python exception in the run method. When this is the case, the state of the underlying process is unknown - it may have been started or not.

Definition at line 211 of file model.py.

6.32.3.5 from_data()

```
def codar.savanna.model.Run.from_data (
    cls,
    data )
```

Create Run instance from nested dictionary data structure, e.g. parsed from JSON. The keys 'name', 'exe', 'args' are required, all the other keys are optional and have the same names as the constructor args. Raises KeyError if a required key is missing.

Definition at line 146 of file model.py.

6.32.3.6 `get_nodes_used()`

```
def codar.savanna.model.Run.get_nodes_used (
    self )
```

Get number of nodes needed to run this app. Requires that the pipeline `set_ppn` method has been called to set this and `tasks_per_node` on each run.

Definition at line 436 of file `model.py`.

6.32.3.7 `get_pid()`

```
def codar.savanna.model.Run.get_pid (
    self )
```

Definition at line 421 of file `model.py`.

6.32.3.8 `get_returncode()`

```
def codar.savanna.model.Run.get_returncode (
    self )
```

Definition at line 416 of file `model.py`.

6.32.3.9 `join()`

```
def codar.savanna.model.Run.join (
    self )
```

Definition at line 431 of file `model.py`.

6.32.3.10 `kill()`

```
def codar.savanna.model.Run.kill (
    self )
```

Kill process and cause run thread to complete after the wait returns. If the run is already done, does nothing. If the process is killed, it will mark the state as killed so it can be re-run on workflow restart. Thread safe.

Definition at line 319 of file `model.py`.

6.32.3.11 killed()

```
def codar.savanna.model.Run.killed (
    self )
```

True if the run is done and the kill method was called. Note that this will `_NOT_` be true if an external kill signal caused the process to exit. Raises `ValueError` if the run is not complete.

Definition at line 202 of file model.py.

6.32.3.12 mpmc_run()

```
def codar.savanna.model.Run.mpmc_run (
    cls,
    runs )
```

Definition at line 171 of file model.py.

6.32.3.13 remove_callback()

```
def codar.savanna.model.Run.remove_callback (
    self,
    fn )
```

Definition at line 234 of file model.py.

6.32.3.14 run()

```
def codar.savanna.model.Run.run (
    self )
```

Definition at line 237 of file model.py.

6.32.3.15 set_runner()

```
def codar.savanna.model.Run.set_runner (
    self,
    runner )
```

Definition at line 188 of file model.py.

6.32.3.16 succeeded()

```
def codar.savanna.model.Run.succeeded (
    self )
```

True if the run is done, finished normally, and had 0 return value.
Raises ValueError if the run is not complete.

Definition at line 218 of file model.py.

6.32.3.17 timed_out()

```
def codar.savanna.model.Run.timed_out (
    self )
```

True if the run is done and was killed because it exceeded the
specified run timeout. Raises ValueError if the run is not complete.

Definition at line 194 of file model.py.

6.32.4 Member Data Documentation

6.32.4.1 args

```
codar.savanna.model.Run.args
```

Definition at line 78 of file model.py.

6.32.4.2 callbacks

```
codar.savanna.model.Run.callbacks
```

Definition at line 116 of file model.py.

6.32.4.3 depends_on_runs

```
codar.savanna.model.Run.depends_on_runs
```

Definition at line 125 of file model.py.

6.32.4.4 env

`codar.savanna.model.Run.env`

Definition at line 80 of file model.py.

6.32.4.5 erf_file

`codar.savanna.model.Run.erf_file`

Definition at line 139 of file model.py.

6.32.4.6 exe

`codar.savanna.model.Run.exe`

Definition at line 77 of file model.py.

6.32.4.7 hostfile

`codar.savanna.model.Run.hostfile`

Definition at line 128 of file model.py.

6.32.4.8 log_prefix

`codar.savanna.model.Run.log_prefix`

Definition at line 114 of file model.py.

6.32.4.9 machine

`codar.savanna.model.Run.machine`

Definition at line 132 of file model.py.

6.32.4.10 name

`codar.savanna.model.Run.name`

Definition at line 76 of file model.py.

6.32.4.11 node_config

`codar.savanna.model.Run.node_config`

Definition at line 136 of file model.py.

6.32.4.12 nodes

`codar.savanna.model.Run.nodes`

Definition at line 121 of file model.py.

6.32.4.13 nodes_assigned

`codar.savanna.model.Run.nodes_assigned`

Definition at line 133 of file model.py.

6.32.4.14 nprocs

`codar.savanna.model.Run.nprocs`

Definition at line 83 of file model.py.

6.32.4.15 res_set

`codar.savanna.model.Run.res_set`

Definition at line 88 of file model.py.

6.32.4.16 return_path

`codar.savanna.model.Run.return_path`

Definition at line 94 of file model.py.

6.32.4.17 runner

`codar.savanna.model.Run.runner`

Definition at line 115 of file model.py.

6.32.4.18 runner_override

`codar.savanna.model.Run.runner_override`

Definition at line 143 of file model.py.

6.32.4.19 sched_args

`codar.savanna.model.Run.sched_args`

Definition at line 79 of file model.py.

6.32.4.20 sleep_after

`codar.savanna.model.Run.sleep_after`

Definition at line 98 of file model.py.

6.32.4.21 stderr_path

`codar.savanna.model.Run.stderr_path`

Definition at line 92 of file model.py.

6.32.4.22 `stdout_path`

`codar.savanna.model.Run.stdout_path`

Definition at line 90 of file `model.py`.

6.32.4.23 `tasks_per_node`

`codar.savanna.model.Run.tasks_per_node`

Definition at line 122 of file `model.py`.

6.32.4.24 `timeout`

`codar.savanna.model.Run.timeout`

Definition at line 82 of file `model.py`.

6.32.4.25 `walltime_path`

`codar.savanna.model.Run.walltime_path`

Definition at line 96 of file `model.py`.

6.32.4.26 `working_dir`

`codar.savanna.model.Run.working_dir`

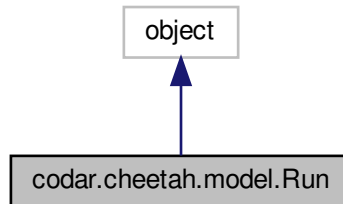
Definition at line 81 of file `model.py`.

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

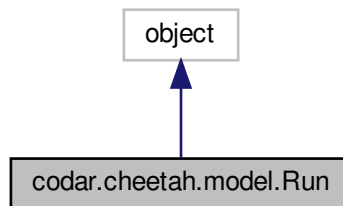
- [savanna/model.py](#)

6.33 codar.cheetah.model.Run Class Reference

Inheritance diagram for codar.cheetah.model.Run:



Collaboration diagram for codar.cheetah.model.Run:



Public Member Functions

- def `__init__` (self, `instance`, `codes`, `codes_path`, `run_path`, `inputs`, `machine`, `node_layout`, `rc_dependency`, `component_subdirs`, `sosflow_profiling`, `sosflow_analysis`, `component_inputs=None`)
- def `get_fob_data_list` (self)
- def `get_total_nprocs` (self)
- def `get_app_param_dict` (self)
- def `add_dataspaces_support` (self, `machine`)
- def `insert_sosflow` (self, `sosd_path`, `sos_analysis_path`, `run_path`, `ppn`)

Public Attributes

- `instance`
- `codes`
- `codes_path`
- `run_path`
- `run_id`

- [inputs](#)
- [machine](#)
- [node_layout](#)
- [component_subdirs](#)
- [sosflow_profiling](#)
- [sosflow_analysis](#)
- [component_inputs](#)
- [total_nodes](#)
- [run_components](#)

6.33.1 Detailed Description

Class representing how to actually run an instance on a given environment, including how to generate arg arrays for executing each code required for the application.

TODO: create a model shared between workflow and cheetah, i.e. `codar.model`

Definition at line 401 of file `model.py`.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 `__init__()`

```
def codar.cheetah.model.Run.__init__ (
    self,
    instance,
    codes,
    codes_path,
    run_path,
    inputs,
    machine,
    node_layout,
    rc_dependency,
    component_subdirs,
    sosflow_profiling,
    sosflow_analyis,
    component_inputs = None )
```

Definition at line 411 of file `model.py`.

6.33.3 Member Function Documentation

6.33.3.1 add_dataspaces_support()

```
def codar.cheetah.model.Run.add_dataspaces_support (
    self,
    machine )

Add support for dataspace.
Check RC Adios xml files to see if any transport methods are marked
for coupling with DATASPACE/DIMES.
For stage_write, check command line args to see if DATASPACE/DIMES
is specified.
:param machine: The current machine. I dont like this here.
:return:
```

Definition at line 630 of file model.py.

6.33.3.2 get_app_param_dict()

```
def codar.cheetah.model.Run.get_app_param_dict (
    self )

Return dictionary containing only the app parameters
(does not include nprocs or exe paths).
```

Definition at line 625 of file model.py.

6.33.3.3 get_fob_data_list()

```
def codar.cheetah.model.Run.get_fob_data_list (
    self )
```

Definition at line 528 of file model.py.

6.33.3.4 get_total_nprocs()

```
def codar.cheetah.model.Run.get_total_nprocs (
    self )
```

Definition at line 545 of file model.py.

6.33.3.5 insert_sosflow()

```
def codar.cheetah.model.Run.insert_sosflow (
    self,
    sosd_path,
    sos_analysis_path,
    run_path,
    ppn )
```

Insert a new component at start of list to launch sosflow daemon.
Should be called only once.

Definition at line 752 of file model.py.

6.33.4 Member Data Documentation

6.33.4.1 codes

```
codar.cheetah.model.Run.codes
```

Definition at line 413 of file model.py.

6.33.4.2 codes_path

```
codar.cheetah.model.Run.codes_path
```

Definition at line 414 of file model.py.

6.33.4.3 component_inputs

```
codar.cheetah.model.Run.component_inputs
```

Definition at line 425 of file model.py.

6.33.4.4 component_subdirs

```
codar.cheetah.model.Run.component_subdirs
```

Definition at line 422 of file model.py.

6.33.4.5 inputs

`codar.cheetah.model.Run.inputs`

Definition at line 417 of file model.py.

6.33.4.6 instance

`codar.cheetah.model.Run.instance`

Definition at line 412 of file model.py.

6.33.4.7 machine

`codar.cheetah.model.Run.machine`

Definition at line 418 of file model.py.

6.33.4.8 node_layout

`codar.cheetah.model.Run.node_layout`

Definition at line 421 of file model.py.

6.33.4.9 run_components

`codar.cheetah.model.Run.run_components`

Definition at line 427 of file model.py.

6.33.4.10 run_id

`codar.cheetah.model.Run.run_id`

Definition at line 416 of file model.py.

6.33.4.11 run_path

`codar.cheetah.model.Run.run_path`

Definition at line 415 of file model.py.

6.33.4.12 sosflow_analysis

`codar.cheetah.model.Run.sosflow_analysis`

Definition at line 424 of file model.py.

6.33.4.13 sosflow_profiling

`codar.cheetah.model.Run.sosflow_profiling`

Definition at line 423 of file model.py.

6.33.4.14 total_nodes

`codar.cheetah.model.Run.total_nodes`

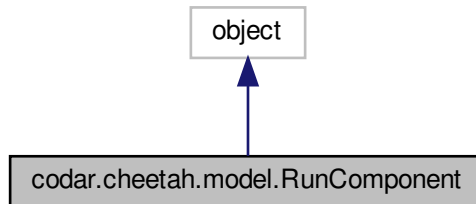
Definition at line 426 of file model.py.

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

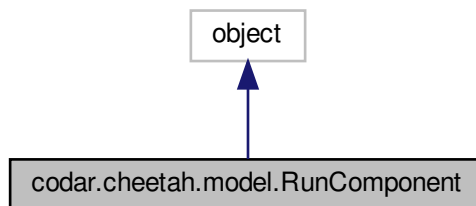
- cheetah/[model.py](#)

6.34 codar.cheetah.model.RunComponent Class Reference

Inheritance diagram for codar.cheetah.model.RunComponent:



Collaboration diagram for codar.cheetah.model.RunComponent:



Public Member Functions

- `def __init__ (self, name, exe, args, sched_args, nprocs, working_dir, component_inputs=None, sleep_after=None, linked_with_sosflow=False, adios_xml_file=None, env=None, timeout=None, hostfile=None, runner_override=False)`
- `def as_fob_data (self)`

Public Attributes

- [name](#)
- [exe](#)
- [args](#)
- [sched_args](#)
- [nprocs](#)
- [sleep_after](#)
- [env](#)
- [timeout](#)

- [working_dir](#)
- [component_inputs](#)
- [linked_with_sosflow](#)
- [adios_xml_file](#)
- [hostfile](#)
- [after_rc_done](#)
- [runner_override](#)

6.34.1 Detailed Description

Definition at line 870 of file model.py.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 `__init__()`

```
def codar.cheetah.model.RunComponent.__init__ (
    self,
    name,
    exe,
    args,
    sched_args,
    nprocs,
    working_dir,
    component_inputs = None,
    sleep_after = None,
    linked_with_sosflow = False,
    adios_xml_file = None,
    env = None,
    timeout = None,
    hostfile = None,
    runner_override = False )
```

Definition at line 874 of file model.py.

6.34.3 Member Function Documentation

6.34.3.1 `as_fob_data()`

```
def codar.cheetah.model.RunComponent.as_fob_data (
    self )
```

Definition at line 891 of file model.py.

6.34.4 Member Data Documentation

6.34.4.1 adios_xml_file

`codar.cheetah.model.RunComponent.adios_xml_file`

Definition at line 886 of file model.py.

6.34.4.2 after_rc_done

`codar.cheetah.model.RunComponent.after_rc_done`

Definition at line 888 of file model.py.

6.34.4.3 args

`codar.cheetah.model.RunComponent.args`

Definition at line 877 of file model.py.

6.34.4.4 component_inputs

`codar.cheetah.model.RunComponent.component_inputs`

Definition at line 884 of file model.py.

6.34.4.5 env

`codar.cheetah.model.RunComponent.env`

Definition at line 881 of file model.py.

6.34.4.6 exe

`codar.cheetah.model.RunComponent.exe`

Definition at line 876 of file model.py.

6.34.4.7 hostfile

`codar.cheetah.model.RunComponent.hostfile`

Definition at line 887 of file model.py.

6.34.4.8 linked_with_sosflow

`codar.cheetah.model.RunComponent.linked_with_sosflow`

Definition at line 885 of file model.py.

6.34.4.9 name

`codar.cheetah.model.RunComponent.name`

Definition at line 875 of file model.py.

6.34.4.10 nprocs

`codar.cheetah.model.RunComponent.nprocs`

Definition at line 879 of file model.py.

6.34.4.11 runner_override

`codar.cheetah.model.RunComponent.runner_override`

Definition at line 889 of file model.py.

6.34.4.12 sched_args

`codar.cheetah.model.RunComponent.sched_args`

Definition at line 878 of file model.py.

6.34.4.13 sleep_after

`codar.cheetah.model.RunComponent.sleep_after`

Definition at line 880 of file model.py.

6.34.4.14 timeout

`codar.cheetah.model.RunComponent.timeout`

Definition at line 882 of file model.py.

6.34.4.15 working_dir

`codar.cheetah.model.RunComponent.working_dir`

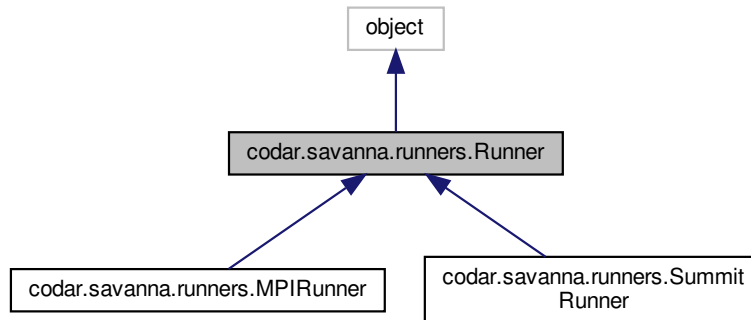
Definition at line 883 of file model.py.

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

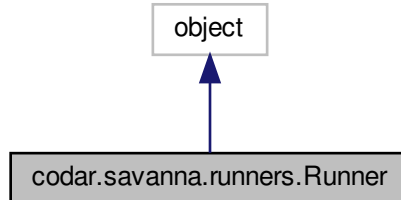
- cheetah/[model.py](#)

6.35 codar.savanna.runners.Runner Class Reference

Inheritance diagram for codar.savanna.runners.Runner:



Collaboration diagram for codar.savanna.runners.Runner:



Public Member Functions

- def `wrap` (self, run, sched_args)

6.35.1 Detailed Description

Definition at line 6 of file `runners.py`.

6.35.2 Member Function Documentation

6.35.2.1 wrap()

```
def codar.savanna.runners.Runner.wrap (
    self,
    run,
    sched_args )
```

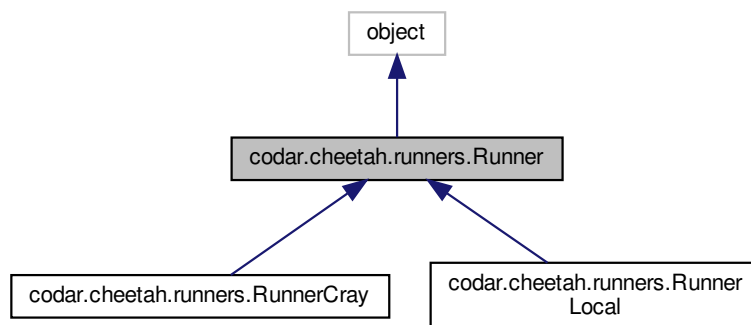
Definition at line 7 of file runners.py.

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

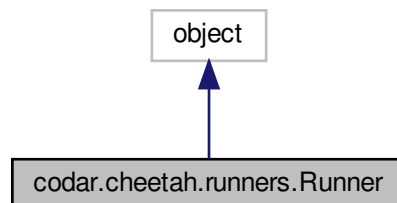
- [savanna/runners.py](#)

6.36 codar.cheetah.runners.Runner Class Reference

Inheritance diagram for codar.cheetah.runners.Runner:



Collaboration diagram for codar.cheetah.runners.Runner:



Public Member Functions

- def [wrap_app_command](#) (self, command_dir, out_name, app_command)

Static Public Attributes

- [name](#) = None

6.36.1 Detailed Description

Definition at line 7 of file runners.py.

6.36.2 Member Function Documentation

6.36.2.1 [wrap_app_command\(\)](#)

```
def codar.cheetah.runners.Runner.wrap_app_command (
    self,
    command_dir,
    out_name,
    app_command )
```

Given an application command line, return a list of commands to run the given line using this runner and in the specified command working directory.

Definition at line 10 of file runners.py.

6.36.3 Member Data Documentation

6.36.3.1 [name](#)

```
codar.cheetah.runners.Runner.name = None [static]
```

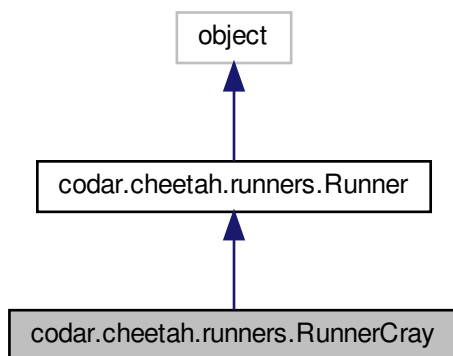
Definition at line 8 of file runners.py.

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

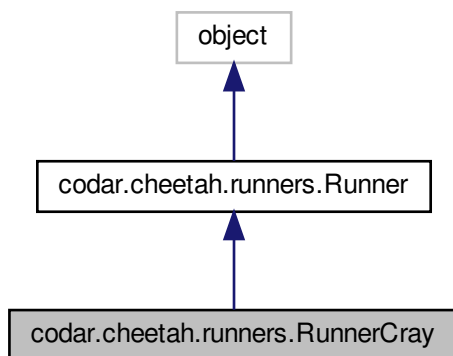
- [cheetah/runners.py](#)

6.37 codar.cheetah.runners.RunnerCray Class Reference

Inheritance diagram for codar.cheetah.runners.RunnerCray:



Collaboration diagram for codar.cheetah.runners.RunnerCray:



Public Member Functions

- def [wrap_app_command](#) (self, command_dir, out_name, app_command)

Static Public Attributes

- string [name](#) = 'cray'

6.37.1 Detailed Description

Definition at line 40 of file runners.py.

6.37.2 Member Function Documentation

6.37.2.1 wrap_app_command()

```
def codar.cheetah.runners.RunnerCray.wrap_app_command (
    self,
    command_dir,
    out_name,
    app_command )
```

Run using aprun, and cd before/after to arrange separate working dir per run.

TODO: how to pass aprun params?

NOTE: assumes CWD is batch directory within the experiment output dir.

Definition at line 43 of file runners.py.

6.37.3 Member Data Documentation

6.37.3.1 name

```
string codar.cheetah.runners.RunnerCray.name = 'cray' [static]
```

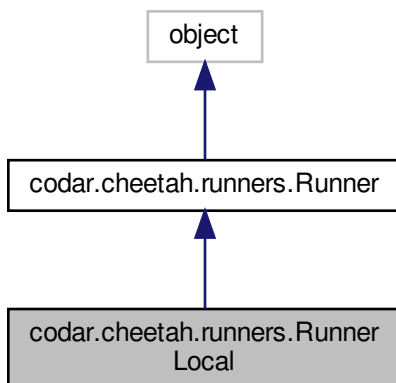
Definition at line 41 of file runners.py.

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

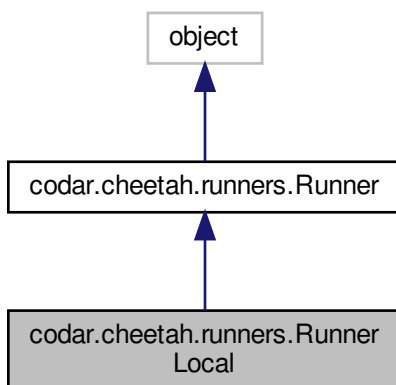
- [cheetah/runners.py](#)

6.38 codar.cheetah.runners.RunnerLocal Class Reference

Inheritance diagram for codar.cheetah.runners.RunnerLocal:



Collaboration diagram for codar.cheetah.runners.RunnerLocal:



Public Member Functions

- def [wrap_app_command](#) (self, command_dir, out_name, app_command)

Static Public Attributes

- string [name](#) = 'local'

6.38.1 Detailed Description

Definition at line 20 of file runners.py.

6.38.2 Member Function Documentation

6.38.2.1 wrap_app_command()

```
def codar.cheetah.runners.RunnerLocal.wrap_app_command (
    self,
    command_dir,
    out_name,
    app_command )
```

Run directly, just at cd before/after to arrange separate working dir per run.

TODO: how to pass runner params?

NOTE: assumes CWD is batch directory within the experiment output dir.

Definition at line 23 of file runners.py.

6.38.3 Member Data Documentation

6.38.3.1 name

```
string codar.cheetah.runners.RunnerLocal.name = 'local' [static]
```

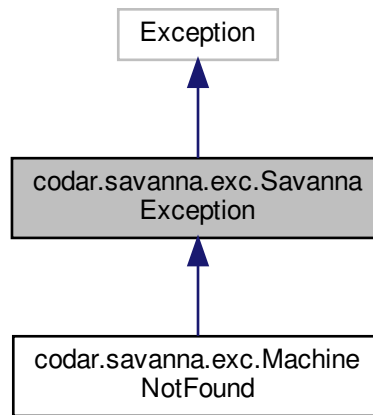
Definition at line 21 of file runners.py.

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

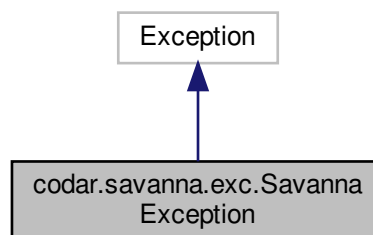
- [cheetah/runners.py](#)

6.39 `codar.savanna.exc.SavannaException` Class Reference

Inheritance diagram for `codar.savanna.exc.SavannaException`:



Collaboration diagram for `codar.savanna.exc.SavannaException`:



6.39.1 Detailed Description

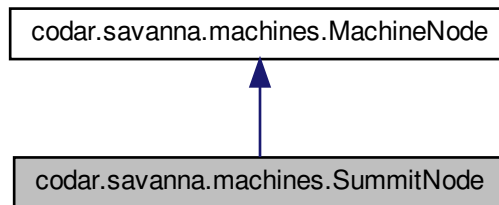
Definition at line 6 of file `exc.py`.

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

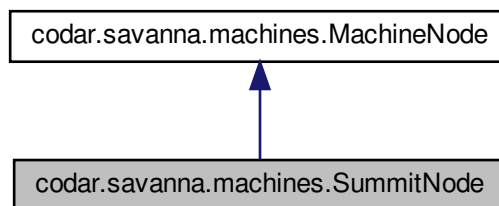
- `savanna/exc.py`

6.40 `codar.savanna.machines.SummitNode` Class Reference

Inheritance diagram for `codar.savanna.machines.SummitNode`:



Collaboration diagram for `codar.savanna.machines.SummitNode`:



Public Member Functions

- def `__init__` (self)
- def `validate_layout` (self)
- def `to_json` (self)

Additional Inherited Members

6.40.1 Detailed Description

Definition at line 28 of file `machines.py`.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 `__init__()`

```
def codar.savanna.machines.SummitNode.__init__ (
    self )
```

Definition at line 29 of file machines.py.

6.40.3 Member Function Documentation

6.40.3.1 `to_json()`

```
def codar.savanna.machines.SummitNode.to_json (
    self )
```

Definition at line 64 of file machines.py.

6.40.3.2 `validate_layout()`

```
def codar.savanna.machines.SummitNode.validate_layout (
    self )
```

Check that 1) the same rank of the same code is not repeated,
2) a gpu is not mapped to multiple executables.

Definition at line 32 of file machines.py.

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

- savanna/[machines.py](#)

6.41 codar.cheetah.parameters.SummitOpts Class Reference

Public Member Functions

- def [__init__](#) (self)

6.41.1 Detailed Description

Definition at line 486 of file parameters.py.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 `__init__()`

```
def codar.cheetah.parameters.SummitOpts.__init__ (
    self )
```

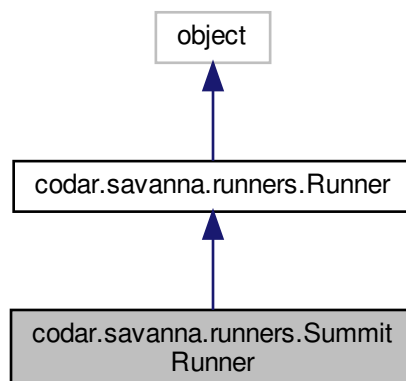
Definition at line 487 of file parameters.py.

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

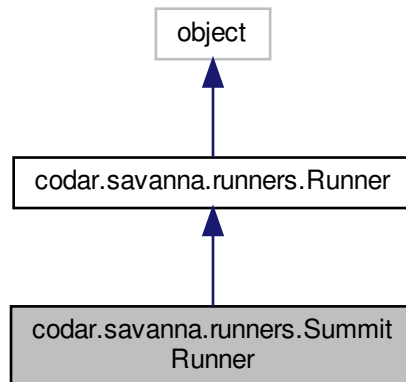
- cheetah/[parameters.py](#)

6.42 `codar.savanna.runners.SummitRunner` Class Reference

Inheritance diagram for `codar.savanna.runners.SummitRunner`:



Collaboration diagram for codar.savanna.runners.SummitRunner:



Public Member Functions

- `def __init__ (self)`
- `def wrap (self, run, sched_args)`
- `def wrap_deprecated (self, run, jsrun_opts, find_in_path=True)`

Public Attributes

- [exe](#)
- [nrs_arg](#)
- [tasks_per_rs_arg](#)
- [cpus_per_rs_arg](#)
- [gpus_per_rs_arg](#)
- [rs_per_host_arg](#)
- [launch_distribution_arg](#)
- [bind_arg](#)
- [machine](#)

6.42.1 Detailed Description

Definition at line 44 of file `runners.py`.

6.42.2 Constructor & Destructor Documentation

6.42.2.1 `__init__()`

```
def codar.savanna.runners.SummitRunner.__init__ (
    self )
```

Definition at line 45 of file runners.py.

6.42.3 Member Function Documentation

6.42.3.1 `wrap()`

```
def codar.savanna.runners.SummitRunner.wrap (
    self,
    run,
    sched_args )
```

Definition at line 56 of file runners.py.

6.42.3.2 `wrap_deprecated()`

```
def codar.savanna.runners.SummitRunner.wrap_deprecated (
    self,
    run,
    jsrun_opts,
    find_in_path = True )
```

This function is deprecated in favor of the above that uses erf files

Definition at line 60 of file runners.py.

6.42.4 Member Data Documentation

6.42.4.1 `bind_arg`

```
codar.savanna.runners.SummitRunner.bind_arg
```

Definition at line 53 of file runners.py.

6.42.4.2 cpus_per_rs_arg

`codar.savanna.runners.SummitRunner.cpus_per_rs_arg`

Definition at line 49 of file runners.py.

6.42.4.3 exe

`codar.savanna.runners.SummitRunner.exe`

Definition at line 46 of file runners.py.

6.42.4.4 gpus_per_rs_arg

`codar.savanna.runners.SummitRunner.gpus_per_rs_arg`

Definition at line 50 of file runners.py.

6.42.4.5 launch_distribution_arg

`codar.savanna.runners.SummitRunner.launch_distribution_arg`

Definition at line 52 of file runners.py.

6.42.4.6 machine

`codar.savanna.runners.SummitRunner.machine`

Definition at line 54 of file runners.py.

6.42.4.7 nrs_arg

`codar.savanna.runners.SummitRunner.nrs_arg`

Definition at line 47 of file runners.py.

6.42.4.8 rs_per_host_arg

`codar.savanna.runners.SummitRunner.rs_per_host_arg`

Definition at line 51 of file runners.py.

6.42.4.9 tasks_per_rs_arg

`codar.savanna.runners.SummitRunner.tasks_per_rs_arg`

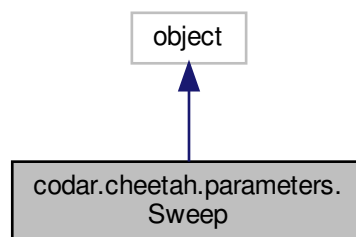
Definition at line 48 of file runners.py.

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

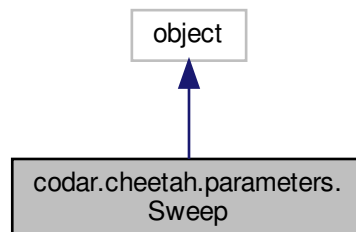
- [savanna/runners.py](#)

6.43 codar.cheetah.parameters.Sweep Class Reference

Inheritance diagram for `codar.cheetah.parameters.Sweep`:



Collaboration diagram for `codar.cheetah.parameters.Sweep`:



Public Member Functions

- def `__init__` (self, `parameters`, `node_layout`=None, `rc_dependency`=None)
- def `get_instances` (self)

Public Attributes

- `parameters`
- `node_layout`
- `rc_dependency`

6.43.1 Detailed Description

Class representing a set of parameter values to search over as a cross product.

Definition at line 45 of file `parameters.py`.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 `__init__()`

```
def codar.cheetah.parameters.Sweep.__init__ (
    self,
    parameters,
    node_layout = None,
    rc_dependency = None )
```

Definition at line 50 of file `parameters.py`.

6.43.3 Member Function Documentation

6.43.3.1 `get_instances()`

```
def codar.cheetah.parameters.Sweep.get_instances (
    self )
```

Get a list of Instance objects representing dense cross product over param values.

TODO: this works great for command line options and args, but what about for config and other types of params? Need to setup a run dir and populate it with filled config templates.

Also how to pass per run output dir? Or is just making CWD the per run dir enough for all cases we care about?

TODO: should have same signature as `SweepGroup` version OR a different name.

Definition at line 55 of file `parameters.py`.

6.43.4 Member Data Documentation

6.43.4.1 `node_layout`

`codar.cheetah.parameters.Sweep.node_layout`

Definition at line 52 of file `parameters.py`.

6.43.4.2 `parameters`

`codar.cheetah.parameters.Sweep.parameters`

Definition at line 51 of file `parameters.py`.

6.43.4.3 `rc_dependency`

`codar.cheetah.parameters.Sweep.rc_dependency`

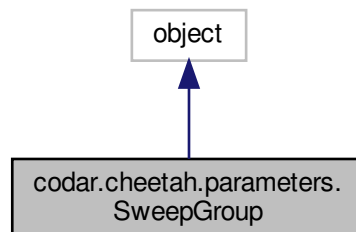
Definition at line 53 of file `parameters.py`.

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

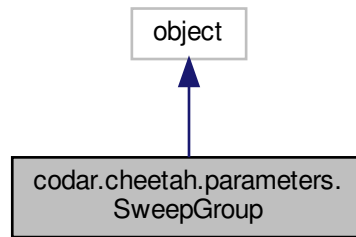
- [cheetah/parameters.py](#)

6.44 `codar.cheetah.parameters.SweepGroup` Class Reference

Inheritance diagram for `codar.cheetah.parameters.SweepGroup`:



Collaboration diagram for codar.cheetah.parameters.SweepGroup:



Public Member Functions

- `def __init__(self, name, parameter_groups, component_subdirs=False, component_inputs=None, walltime=3600, max_procs=None, per_run_timeout=None, sosflow_profiling=False, sosflow_analysis=False, nodes=None, launch_mode=None, run_repetitions=0)`

Public Attributes

- `name`
- `nodes`
- `component_subdirs`
- `max_procs`
- `parameter_groups`
- `walltime`
- `per_run_timeout`
- `sosflow_profiling`
- `sosflow_analysis`
- `component_inputs`
- `launch_mode`
- `run_repetitions`

6.44.1 Detailed Description

Class representing a grouping of run parameters that can be executed by a single scheduler job, because they share the same scheduler parameters.

Note that `nodes` is no longer required - if not specified, it is calculated based on the biggest run within the group.

How this gets converted into a script depends on the target machine and which scheduler (if any) that machine uses.

Definition at line 11 of file `parameters.py`.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 `__init__()`

```
def codar.cheetah.parameters.SweepGroup.__init__ (
    self,
    name,
    parameter_groups,
    component_subdirs = False,
    component_inputs = None,
    walltime = 3600,
    max_procs = None,
    per_run_timeout = None,
    sosflow_profiling = False,
    sosflow_analysis = False,
    nodes = None,
    launch_mode = None,
    run_repetitions = 0 )
```

Definition at line 26 of file parameters.py.

6.44.3 Member Data Documentation

6.44.3.1 `component_inputs`

```
codar.cheetah.parameters.SweepGroup.component_inputs
```

Definition at line 37 of file parameters.py.

6.44.3.2 `component_subdirs`

```
codar.cheetah.parameters.SweepGroup.component_subdirs
```

Definition at line 29 of file parameters.py.

6.44.3.3 `launch_mode`

```
codar.cheetah.parameters.SweepGroup.launch_mode
```

Definition at line 41 of file parameters.py.

6.44.3.4 max_procs

`codar.cheetah.parameters.SweepGroup.max_procs`

Definition at line 30 of file parameters.py.

6.44.3.5 name

`codar.cheetah.parameters.SweepGroup.name`

Definition at line 27 of file parameters.py.

6.44.3.6 nodes

`codar.cheetah.parameters.SweepGroup.nodes`

Definition at line 28 of file parameters.py.

6.44.3.7 parameter_groups

`codar.cheetah.parameters.SweepGroup.parameter_groups`

Definition at line 31 of file parameters.py.

6.44.3.8 per_run_timeout

`codar.cheetah.parameters.SweepGroup.per_run_timeout`

Definition at line 34 of file parameters.py.

6.44.3.9 run_repetitions

`codar.cheetah.parameters.SweepGroup.run_repetitions`

Definition at line 42 of file parameters.py.

6.44.3.10 `sosflow_analysis`

`codar.cheetah.parameters.SweepGroup.sosflow_analysis`

Definition at line 36 of file `parameters.py`.

6.44.3.11 `sosflow_profiling`

`codar.cheetah.parameters.SweepGroup.sosflow_profiling`

Definition at line 35 of file `parameters.py`.

6.44.3.12 `walltime`

`codar.cheetah.parameters.SweepGroup.walltime`

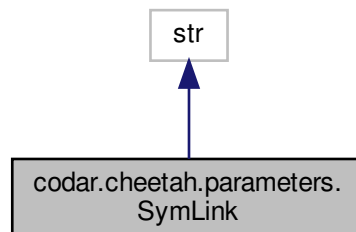
Definition at line 32 of file `parameters.py`.

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

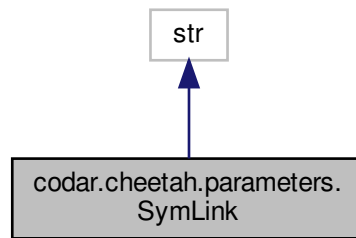
- [cheetah/parameters.py](#)

6.45 `codar.cheetah.parameters.SymLink` Class Reference

Inheritance diagram for `codar.cheetah.parameters.SymLink`:



Collaboration diagram for codar.cheetah.parameters.SymLink:



Public Member Functions

- `def __init__(self, source)`

Public Attributes

- [source](#)

6.45.1 Detailed Description

Class to represent symbolic links as an input type for a run component

Definition at line 491 of file `parameters.py`.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 `__init__()`

```
def codar.cheetah.parameters.SymLink.__init__ (
    self,
    source )
```

Definition at line 495 of file `parameters.py`.

6.45.3 Member Data Documentation

6.45.3.1 source

`codar.cheetah.parameters.SymLink.source`

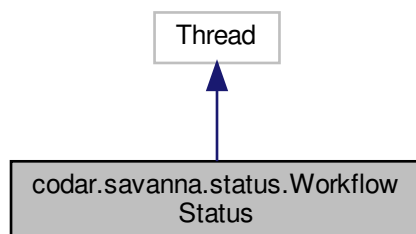
Definition at line 496 of file `parameters.py`.

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

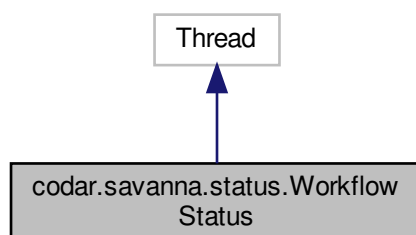
- cheetah/[parameters.py](#)

6.46 `codar.savanna.status.WorkflowStatus` Class Reference

Inheritance diagram for `codar.savanna.status.WorkflowStatus`:



Collaboration diagram for `codar.savanna.status.WorkflowStatus`:



Public Member Functions

- `def __init__(self, file_path)`
- `def set_state(self, pipeline_state)`

Public Attributes

- [file_path](#)

6.46.1 Detailed Description

Definition at line 24 of file status.py.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 `__init__()`

```
def codar.savanna.status.WorkflowStatus.__init__ (
    self,
    file_path )
```

Definition at line 25 of file status.py.

6.46.3 Member Function Documentation

6.46.3.1 `set_state()`

```
def codar.savanna.status.WorkflowStatus.set_state (
    self,
    pipeline_state )
```

Definition at line 37 of file status.py.

6.46.4 Member Data Documentation

6.46.4.1 `file_path`

```
codar.savanna.status.WorkflowStatus.file_path
```

Definition at line 27 of file status.py.

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

- savanna/[status.py](#)

Chapter 7

File Documentation

7.1 `__init__.py` File Reference

Namespaces

- [codar](#)

7.2 `cheetah/__init__.py` File Reference

Namespaces

- [codar.cheetah](#)

7.3 `savanna/__init__.py` File Reference

Namespaces

- [codar.savanna](#)

7.4 `cheetah/adios2_interface.py` File Reference

Namespaces

- [codar.cheetah.adios2_interface](#)

Functions

- def [codar.cheetah.adios2_interface.get_adios_version](#) (xml_file)
- def [codar.cheetah.adios2_interface.set_engine](#) (xmlfile, io_obj, engine_type, parameters=None)
- def [codar.cheetah.adios2_interface.set_transport](#) (xmlfile, io_obj, transport_type, parameters=None)
- def [codar.cheetah.adios2_interface.set_var_operation](#) (xmlfile, io_obj, var_name, operation, parameters=None)

7.5 cheetah/adios_params.py File Reference

Namespaces

- [codar.cheetah.adios_params](#)

Functions

- def [codar.cheetah.adios_params.adios_xml_transform](#) (xml_filepath, group_name, var_name, value)
- def [codar.cheetah.adios_params.adios_xml_transport](#) (xml_filepath, group_name, method_name, method_opts)
- def [codar.cheetah.adios_params.xml_has_transport](#) (xml_filepath, transport_type)

7.6 cheetah/config.py File Reference

Namespaces

- [codar.cheetah.config](#)

Functions

- def [codar.cheetah.config.scheduler_path](#) (scheduler_name)
- def [codar.cheetah.config.machine_submit_env_path](#) (machine_name)
- def [codar.cheetah.config.etc_path](#) (conf_name)
- def [codar.cheetah.config.get_dataspaces_num_servers](#) (num_dimes_clients, num_dataspaces_clients)

Variables

- [codar.cheetah.config.PACKAGE_PATH](#) = os.path.realpath(os.path.dirname(__file__))
- [codar.cheetah.config.DATA_PATH](#) = os.path.join(PACKAGE_PATH, "data")
- [codar.cheetah.config.CODAR_PATH](#) = os.path.realpath(os.path.join(PACKAGE_PATH, ".."))
- [codar.cheetah.config.CHEETAH_PATH_SCHEDULER](#) = os.path.join(DATA_PATH, "scheduler")
- [codar.cheetah.config.CHEETAH_PATH_MACHINE_CONFIG](#) = os.path.join(DATA_PATH, "machine_config")
- [codar.cheetah.config.WORKFLOW_SCRIPT](#) = os.path.join(CODAR_PATH, "savanna", "main.py")

7.7 cheetah/exc.py File Reference

Classes

- class [codar.cheetah.exc.CheetahException](#)
- class [codar.cheetah.exc.MachineNotFound](#)
- class [codar.cheetah.exc.CampaignParseError](#)

Namespaces

- [codar.cheetah.exc](#)

7.8 savanna/exc.py File Reference

Classes

- class [codar.savanna.exc.SavannaException](#)
- class [codar.savanna.exc.MachineNotFound](#)

Namespaces

- [codar.savanna.exc](#)

7.9 cheetah/helpers.py File Reference

Namespaces

- [codar.cheetah.helpers](#)

Functions

- def [codar.cheetah.helpers.make_executable](#) (path)
- def [codar.cheetah.helpers.swift_escape_string](#) (s)
- def [codar.cheetah.helpers.parse_timedelta_seconds](#) (v)
- def [codar.cheetah.helpers.copy_to_dir](#) (source_file, dest_dir, follow_symlinks=True)
- def [codar.cheetah.helpers.copy_to_path](#) (source_file, dest_file, follow_symlinks=True)
- def [codar.cheetah.helpers.is_executable](#) (fpath)
- def [codar.cheetah.helpers.copypath_to_dir](#) (source_dir, dest_dir, follow_symlinks=True)
- def [codar.cheetah.helpers.relative_or_absolute_path](#) (prefix, path)
- def [codar.cheetah.helpers.relative_or_absolute_path_list](#) (prefix, path_list)
- def [codar.cheetah.helpers.get_immediate_subdirs](#) (dir_path)
- def [codar.cheetah.helpers.dir_size](#) (path)
- def [codar.cheetah.helpers.get_file_size](#) (dir_entry)
- def [codar.cheetah.helpers.is_campaign_directory](#) (path)
- def [codar.cheetah.helpers.require_campaign_directory](#) (path)
- def [codar.cheetah.helpers.json_config_set_option](#) (filename, key, value)

7.10 cheetah/launchers.py File Reference

Classes

- class [codar.cheetah.launchers.Launcher](#)

Namespaces

- [codar.cheetah.launchers](#)

Variables

- string [codar.cheetah.launchers.TAU_PROFILE_PATTERN](#) = "codar.cheetah.tau-{code}"

7.11 cheetah/loader.py File Reference

Namespaces

- [codar.cheetah.loader](#)

Functions

- def [codar.cheetah.loader.load_experiment_class](#) (file_path)

7.12 cheetah/machine_launchers.py File Reference

Namespaces

- [codar.cheetah.machine_launchers](#)

Functions

- def [codar.cheetah.machine_launchers.get_launcher](#) (machine, output_directory, num_codes)

Variables

- [codar.cheetah.machine_launchers.machine_launchers](#) = dict()

7.13 cheetah/model.py File Reference

Classes

- class [codar.cheetah.model.Campaign](#)
- class [codar.cheetah.model.Run](#)
- class [codar.cheetah.model.RunComponent](#)

Namespaces

- [codar.cheetah.model](#)

Variables

- `codar.cheetah.model.RESERVED_CODE_NAMES` = set(['post-process'])

7.14 savanna/model.py File Reference

Classes

- class `codar.savanna.model.NodeConfig`
- class `codar.savanna.model.Run`
- class `codar.savanna.model.Pipeline`

Namespaces

- `codar.savanna.model`

Variables

- string `codar.savanna.model.STDOUT_NAME` = 'codar.workflow.stdout'
- string `codar.savanna.model.STDERR_NAME` = 'codar.workflow.stderr'
- string `codar.savanna.model.RETURN_NAME` = 'codar.workflow.return'
- string `codar.savanna.model.WALLTIME_NAME` = 'codar.workflow.walltime'
- int `codar.savanna.model.KILL_WAIT` = 30
- int `codar.savanna.model.WAIT_DELAY_KILL` = 30
- int `codar.savanna.model.WAIT_DELAY_GIVE_UP` = 120

7.15 cheetah/parameters.py File Reference

Classes

- class `codar.cheetah.parameters.SweepGroup`
- class `codar.cheetah.parameters.Sweep`
- class `codar.cheetah.parameters.ParameterValue`
- class `codar.cheetah.parameters.Instance`
- class `codar.cheetah.parameters.CodeCommand`
- class `codar.cheetah.parameters.Param`
- class `codar.cheetah.parameters.ParamCmdLineArg`
- class `codar.cheetah.parameters.ParamAdiosXML`
- class `codar.cheetah.parameters.ParamADIOS2XML`
- class `codar.cheetah.parameters.ParamConfig`
- class `codar.cheetah.parameters.ParamKeyValue`
- class `codar.cheetah.parameters.ParamCmdLineOption`
- class `codar.cheetah.parameters.ParamEnvVar`
- class `codar.cheetah.parameters.ParamSchedulerArgs`
- class `codar.cheetah.parameters.ParamRunner`
- class `codar.cheetah.parameters.SummitOpts`
- class `codar.cheetah.parameters.SymLink`

Namespaces

- [codar.cheetah.parameters](#)

7.16 cheetah/pbs.py File Reference

Namespaces

- [codar.cheetah.pbs](#)

Functions

- def [codar.cheetah.pbs.open_pbs_file](#) (scheduler_dir_path, name, project, nodes, walltime)
- def [codar.cheetah.pbs.write_run_script](#) (script_out_path, scheduler_dir_path)

Variables

- string [codar.cheetah.pbs.PBS_NAME](#) = 'job.pbs'
- string [codar.cheetah.pbs.PBS_FORMAT_TEMPLATE](#)
- string [codar.cheetah.pbs.SUBMIT_FORMAT_TEMPLATE](#)

7.17 cheetah/report_generator.py File Reference

Classes

- class [codar.cheetah.report_generator._RunParser](#)
- class [codar.cheetah.report_generator._ReportGenerator](#)

Namespaces

- [codar.cheetah.report_generator](#)

Functions

- def [codar.cheetah.report_generator.generate_report](#) (campaign_directory, user_run_script, output_file_path)

7.18 cheetah/runners.py File Reference

Classes

- class [codar.cheetah.runners.Runner](#)
- class [codar.cheetah.runners.RunnerLocal](#)
- class [codar.cheetah.runners.RunnerCray](#)

Namespaces

- [codar.cheetah.runners](#)

7.19 savanna/runners.py File Reference

Classes

- class [codar.savanna.runners.Runner](#)
- class [codar.savanna.runners.MPIRunner](#)
- class [codar.savanna.runners.SummitRunner](#)

Namespaces

- [codar.savanna.runners](#)

Variables

- [codar.savanna.runners.mpiexec](#) = MPIRunner('mpiexec', '-n', hostfile='--hostfile')
- [codar.savanna.runners.aprun](#) = MPIRunner('aprun', '-n', tasks_per_node_arg='-N', hostfile='-L')
- [codar.savanna.runners.srun](#) = MPIRunner('srun', '-n', nodes_arg='-N', hostfile='-w')
- [codar.savanna.runners.jsrun](#) = SummitRunner()

7.20 cheetah/status.py File Reference

Namespaces

- [codar.cheetah.status](#)

Functions

- def [codar.cheetah.status.print_campaign_status](#) (campaign_directory, filter_user=None, filter_group=None, filter_run=None, filter_code=None, group_summary=False, run_summary=False, print_logs=False, log_level='DEBUG', return_codes=False, print_output=False, show_parameters=False)
- def [codar.cheetah.status.get_workflow_status](#) (status_file_path, print_counts=False, indent=0, print_return_codes=False, filter_run=None, print_parameters=False, filter_code=None, run_summary=False, code_names=None)

7.21 savanna/status.py File Reference

Classes

- class [codar.savanna.status.WorkflowStatus](#)
- class [codar.savanna.status.PipelineState](#)

Namespaces

- [codar.savanna.status](#)

Variables

- string [codar.savanna.status.NOT_STARTED](#) = 'not_started'
- string [codar.savanna.status.RUNNING](#) = 'running'
- string [codar.savanna.status.DONE](#) = 'done'
- string [codar.savanna.status.KILLED](#) = 'killed'
- string [codar.savanna.status.REASON_TIMEOUT](#) = 'timeout'
- string [codar.savanna.status.REASON_FAILED](#) = 'failed'
- string [codar.savanna.status.REASON_SUCCEEDED](#) = 'succeeded'
- string [codar.savanna.status.REASON_EXCEPTION](#) = 'exception'
- string [codar.savanna.status.REASON_NOFIT](#) = 'nofit'

7.22 cheetah/templates.py File Reference

Namespaces

- [codar.cheetah.templates](#)

Variables

- string [codar.cheetah.templates.CAMPAIGN_ENV_TEMPLATE](#)
- string [codar.cheetah.templates.GROUP_ENV_TEMPLATE](#)

7.23 savanna/consumer.py File Reference

Classes

- class [codar.savanna.consumer.PipelineRunner](#)

Namespaces

- [codar.savanna.consumer](#)

7.24 savanna/machines.py File Reference

Classes

- class [codar.savanna.machines.MachineNode](#)
- class [codar.savanna.machines.SummitNode](#)
- class [codar.savanna.machines.Machine](#)

Namespaces

- [codar.savanna.machines](#)

Functions

- def [codar.savanna.machines.get_by_name](#) (name)

Variables

- [codar.savanna.machines.SCHEDULER_OPTIONS](#) = set(["project", "queue", "constraint", "license"])
- [codar.savanna.machines.local](#) = Machine('local', "local", "mpiexec", MachineNode, processes_per_node=1)
- [codar.savanna.machines.titan](#)
- [codar.savanna.machines.cori](#)
- [codar.savanna.machines.theta](#)
- [codar.savanna.machines.summit](#)

7.25 savanna/main.py File Reference

Namespaces

- [codar.savanna.main](#)

Functions

- def [codar.savanna.main.parse_args](#) ()
- def [codar.savanna.main.main](#) ()
- def [codar.savanna.main.get_job_id](#) ()

Variables

- [codar.savanna.main.consumer](#) = None

7.26 savanna/node_layout.py File Reference

Classes

- class [codar.savanna.node_layout.NodeLayout](#)

Namespaces

- [codar.savanna.node_layout](#)

7.27 savanna/producer.py File Reference

Classes

- class [codar.savanna.producer.JSONFilePipelineReader](#)

Namespaces

- [codar.savanna.producer](#)

7.28 savanna/scheduler.py File Reference

Classes

- class [codar.savanna.scheduler.JobList](#)

Namespaces

- [codar.savanna.scheduler](#)

7.29 savanna/summit_helper.py File Reference

Namespaces

- [codar.savanna.summit_helper](#)

Functions

- def [codar.savanna.summit_helper.get_nodes_reqd](#) (res_set, nrs)
- def [codar.savanna.summit_helper.create_erf_file](#) (run)

Index

- `__get__`
 - `codar::cheetah::parameters::Param`, 91
- `__getattr__`
 - `codar::cheetah::parameters::ParameterValue`, 107
- `__init__`
 - `codar::cheetah::exc::MachineNotFound`, 80
 - `codar::cheetah::launchers::Launcher`, 68
 - `codar::cheetah::model::Campaign`, 49
 - `codar::cheetah::model::Run`, 142
 - `codar::cheetah::model::RunComponent`, 148
 - `codar::cheetah::parameters::CodeCommand`, 57
 - `codar::cheetah::parameters::Instance`, 60
 - `codar::cheetah::parameters::Param`, 91
 - `codar::cheetah::parameters::ParamADIOS2XML`, 94
 - `codar::cheetah::parameters::ParamAdiosXML`, 97
 - `codar::cheetah::parameters::ParamCmdLineArg`, 99
 - `codar::cheetah::parameters::ParamCmdLineOption`, 101
 - `codar::cheetah::parameters::ParamConfig`, 103
 - `codar::cheetah::parameters::ParamEnvVar`, 105
 - `codar::cheetah::parameters::ParamKeyValue`, 109
 - `codar::cheetah::parameters::ParamRunner`, 111
 - `codar::cheetah::parameters::ParamSchedulerArgs`, 113
 - `codar::cheetah::parameters::ParameterValue`, 107
 - `codar::cheetah::parameters::SummitOpts`, 162
 - `codar::cheetah::parameters::Sweep`, 167
 - `codar::cheetah::parameters::SweepGroup`, 170
 - `codar::cheetah::parameters::SymLink`, 173
 - `codar::cheetah::report_generator::_ReportGenerator`, 39
 - `codar::cheetah::report_generator::_RunParser`, 43
 - `codar::savanna::consumer::PipelineRunner`, 123
 - `codar::savanna::exc::MachineNotFound`, 79
 - `codar::savanna::machines::Machine`, 73
 - `codar::savanna::machines::MachineNode`, 76
 - `codar::savanna::machines::SummitNode`, 160
 - `codar::savanna::model::NodeConfig`, 84
 - `codar::savanna::model::Pipeline`, 115
 - `codar::savanna::model::Run`, 132
 - `codar::savanna::node_layout::NodeLayout`, 86
 - `codar::savanna::producer::JSONFilePipelineReader`, 66
 - `codar::savanna::runners::MPIRunner`, 82
 - `codar::savanna::runners::SummitRunner`, 163
 - `codar::savanna::scheduler::JobList`, 64
 - `codar::savanna::status::PipelineState`, 129
 - `codar::savanna::status::WorkflowStatus`, 175
- `__init__.py`, 177
- `__len__`
 - `codar::cheetah::parameters::Param`, 92
 - `codar::savanna::scheduler::JobList`, 64
- `add_arg`
 - `codar::cheetah::parameters::CodeCommand`, 57
- `add_callback`
 - `codar::savanna::model::Run`, 132
- `add_dataspaces_support`
 - `codar::cheetah::model::Run`, 142
- `add_done_callback`
 - `codar::savanna::model::Pipeline`, 115
- `add_fatal_callback`
 - `codar::savanna::model::Pipeline`, 115
- `add_job`
 - `codar::savanna::scheduler::JobList`, 64
- `add_node`
 - `codar::savanna::node_layout::NodeLayout`, 86
- `add_option`
 - `codar::cheetah::parameters::CodeCommand`, 57
- `add_parameter`
 - `codar::cheetah::parameters::Instance`, 60
- `add_pipeline`
 - `codar::savanna::consumer::PipelineRunner`, 123
- `adios_xml_file`
 - `codar::cheetah::model::RunComponent`, 149
- `adios_xml_transform`
 - `codar::cheetah::adios_params`, 12
- `adios_xml_transport`
 - `codar::cheetah::adios_params`, 12
- `after_rc_done`
 - `codar::cheetah::model::RunComponent`, 149
- `allocated_nodes`
 - `codar::savanna::consumer::PipelineRunner`, 125
- `app_config_scripts`
 - `codar::cheetah::model::Campaign`, 49
- `app_dir`
 - `codar::cheetah::model::Campaign`, 50
- `aprun`
 - `codar::savanna::runners`, 35
- `args`
 - `codar::cheetah::model::RunComponent`, 149
 - `codar::cheetah::parameters::CodeCommand`, 58
 - `codar::savanna::model::Run`, 136
- `as_data`
 - `codar::savanna::status::PipelineState`, 129
- `as_data_list`
 - `codar::savanna::node_layout::NodeLayout`, 86

- as_dict
 - codar::cheetah::parameters::Instance, 60
- as_fob_data
 - codar::cheetah::model::RunComponent, 148
- as_string
 - codar::cheetah::parameters::Instance, 60
- batch_script_name
 - codar::cheetah::launchers::Launcher, 69
- batch_walltime_name
 - codar::cheetah::launchers::Launcher, 69
- bind_arg
 - codar::savanna::runners::SummitRunner, 164
- CAMPAIGN_ENV_TEMPLATE
 - codar::cheetah::templates, 27
- CHEETAH_PATH_MACHINE_CONFIG
 - codar::cheetah::config, 14
- CHEETAH_PATH_SCHEDULER
 - codar::cheetah::config, 14
- CODAR_PATH
 - codar::cheetah::config, 14
- callbacks
 - codar::savanna::model::Run, 136
- campaign_directory
 - codar::cheetah::report_generator::_ReportGenerator, 41
- cheetah/__init__.py, 177
- cheetah/adios2_interface.py, 177
- cheetah/adios_params.py, 178
- cheetah/config.py, 178
- cheetah/exc.py, 178
- cheetah/helpers.py, 179
- cheetah/launchers.py, 179
- cheetah/loader.py, 180
- cheetah/machine_launchers.py, 180
- cheetah/model.py, 180
- cheetah/parameters.py, 181
- cheetah/pbs.py, 182
- cheetah/report_generator.py, 182
- cheetah/runners.py, 182
- cheetah/status.py, 183
- cheetah/templates.py, 184
- close
 - codar::savanna::model::Run, 132
- codar, 9
- codar.cheetah, 9
- codar.cheetah.adios2_interface, 10
- codar.cheetah.adios_params, 11
- codar.cheetah.config, 13
- codar.cheetah.exc, 15
- codar.cheetah.exc.CampaignParseError, 54
- codar.cheetah.exc.CheetahException, 55
- codar.cheetah.exc.MachineNotFound, 79
- codar.cheetah.helpers, 15
- codar.cheetah.launchers, 19
- codar.cheetah.launchers.Launcher, 67
- codar.cheetah.loader, 20
- codar.cheetah.machine_launchers, 21
- codar.cheetah.model, 21
- codar.cheetah.model.Campaign, 47
- codar.cheetah.model.Run, 141
- codar.cheetah.model.RunComponent, 147
- codar.cheetah.parameters, 22
- codar.cheetah.parameters.CodeCommand, 56
- codar.cheetah.parameters.Instance, 59
- codar.cheetah.parameters.Param, 90
- codar.cheetah.parameters.ParamADIOS2XML, 93
- codar.cheetah.parameters.ParamAdiosXML, 96
- codar.cheetah.parameters.ParamCmdLineArg, 98
- codar.cheetah.parameters.ParamCmdLineOption, 100
- codar.cheetah.parameters.ParamConfig, 102
- codar.cheetah.parameters.ParamEnvVar, 104
- codar.cheetah.parameters.ParamKeyValue, 108
- codar.cheetah.parameters.ParamRunner, 110
- codar.cheetah.parameters.ParamSchedulerArgs, 112
- codar.cheetah.parameters.ParameterValue, 106
- codar.cheetah.parameters.SummitOpts, 161
- codar.cheetah.parameters.Sweep, 166
- codar.cheetah.parameters.SweepGroup, 168
- codar.cheetah.parameters.SymLink, 172
- codar.cheetah.pbs, 23
- codar.cheetah.report_generator, 25
- codar.cheetah.report_generator._ReportGenerator, 39
- codar.cheetah.report_generator._RunParser, 42
- codar.cheetah.runners, 25
- codar.cheetah.runners.Runner, 153
- codar.cheetah.runners.RunnerCray, 155
- codar.cheetah.runners.RunnerLocal, 157
- codar.cheetah.status, 26
- codar.cheetah.templates, 27
- codar.savanna, 28
- codar.savanna.consumer, 28
- codar.savanna.consumer.PipelineRunner, 122
- codar.savanna.exc, 28
- codar.savanna.exc.MachineNotFound, 78
- codar.savanna.exc.SavannaException, 159
- codar.savanna.machines, 29
- codar.savanna.machines.Machine, 72
- codar.savanna.machines.MachineNode, 76
- codar.savanna.machines.SummitNode, 160
- codar.savanna.main, 31
- codar.savanna.model, 32
- codar.savanna.model.NodeConfig, 83
- codar.savanna.model.Pipeline, 113
- codar.savanna.model.Run, 130
- codar.savanna.node_layout, 34
- codar.savanna.node_layout.NodeLayout, 85
- codar.savanna.producer, 34
- codar.savanna.producer.JSONFilePipelineReader, 65
- codar.savanna.runners, 34
- codar.savanna.runners.MPIRunner, 81
- codar.savanna.runners.Runner, 152
- codar.savanna.runners.SummitRunner, 162
- codar.savanna.scheduler, 35
- codar.savanna.scheduler.JobList, 63
- codar.savanna.status, 36

- `codar.savanna.status.PipelineState`, 128
- `codar.savanna.status.WorkflowStatus`, 174
- `codar.savanna.summit_helper`, 38
- `codar::cheetah::adios2_interface`
 - `get_adios_version`, 10
 - `set_engine`, 10
 - `set_transport`, 10
 - `set_var_operation`, 11
- `codar::cheetah::adios_params`
 - `adios_xml_transform`, 12
 - `adios_xml_transport`, 12
 - `xml_has_transport`, 12
- `codar::cheetah::config`
 - `CHEETAH_PATH_MACHINE_CONFIG`, 14
 - `CHEETAH_PATH_SCHEDULER`, 14
 - `CODAR_PATH`, 14
 - `DATA_PATH`, 14
 - `etc_path`, 13
 - `get_dataspaces_num_servers`, 13
 - `machine_submit_env_path`, 13
 - `PACKAGE_PATH`, 15
 - `scheduler_path`, 14
 - `WORKFLOW_SCRIPT`, 15
- `codar::cheetah::exc::MachineNotFound`
 - `__init__`, 80
- `codar::cheetah::helpers`
 - `copy_to_dir`, 16
 - `copy_to_path`, 16
 - `copytree_to_dir`, 16
 - `dir_size`, 16
 - `get_file_size`, 17
 - `get_immediate_subdirs`, 17
 - `is_campaign_directory`, 17
 - `is_executable`, 17
 - `json_config_set_option`, 18
 - `make_executable`, 18
 - `parse_timedelta_seconds`, 18
 - `relative_or_absolute_path`, 18
 - `relative_or_absolute_path_list`, 19
 - `require_campaign_directory`, 19
 - `swift_escape_string`, 19
- `codar::cheetah::launchers`
 - `TAU_PROFILE_PATTERN`, 20
- `codar::cheetah::launchers::Launcher`
 - `__init__`, 68
 - `batch_script_name`, 69
 - `batch_walltime_name`, 69
 - `create_group_directory`, 68
 - `jobid_file_name`, 70
 - `machine_name`, 70
 - `name`, 70
 - `num_codes`, 70
 - `output_directory`, 70
 - `read_jobid`, 69
 - `run_command_name`, 70
 - `run_json_name`, 71
 - `run_out_name`, 71
 - `runner_name`, 71
 - `scheduler_name`, 71
 - `status_script_name`, 71
 - `submit_out_name`, 71
 - `submit_script_name`, 72
 - `wait_script_name`, 72
- `codar::cheetah::loader`
 - `load_experiment_class`, 20
- `codar::cheetah::machine_launchers`
 - `get_launcher`, 21
 - `machine_launchers`, 21
- `codar::cheetah::model`
 - `RESERVED_CODE_NAMES`, 22
- `codar::cheetah::model::Campaign`
 - `__init__`, 49
 - `app_config_scripts`, 49
 - `app_dir`, 50
 - `codes`, 50
 - `inputs`, 50
 - `kill_on_partial_failure`, 50
 - `machine`, 51
 - `machine_app_config_script`, 51
 - `machine_scheduler_options`, 51
 - `make_experiment_run_dir`, 49
 - `name`, 51
 - `post_process_script`, 51
 - `python_path`, 51
 - `run_dir_setup_script`, 52
 - `run_post_process_script`, 52
 - `run_post_process_stop_group_on_failure`, 52
 - `runs`, 52
 - `scheduler_options`, 52
 - `sos_analysis_path`, 52
 - `sosd_num_aggregators`, 53
 - `sosd_path`, 53
 - `supported_machines`, 53
 - `sweeps`, 53
 - `tau_config`, 53
 - `umask`, 53
- `codar::cheetah::model::Run`
 - `__init__`, 142
 - `add_dataspaces_support`, 142
 - `codes`, 144
 - `codes_path`, 144
 - `component_inputs`, 144
 - `component_subdirs`, 144
 - `get_app_param_dict`, 143
 - `get_fob_data_list`, 143
 - `get_total_nprocs`, 143
 - `inputs`, 144
 - `insert_sosflow`, 143
 - `instance`, 145
 - `machine`, 145
 - `node_layout`, 145
 - `run_components`, 145
 - `run_id`, 145
 - `run_path`, 145
 - `sosflow_analysis`, 146
 - `sosflow_profiling`, 146

- total_nodes, 146
- codar::cheetah::model::RunComponent
 - __init__, 148
 - adios_xml_file, 149
 - after_rc_done, 149
 - args, 149
 - as_fob_data, 148
 - component_inputs, 149
 - env, 149
 - exe, 149
 - hostfile, 150
 - linked_with_sosflow, 150
 - name, 150
 - nprocs, 150
 - runner_override, 150
 - sched_args, 150
 - sleep_after, 151
 - timeout, 151
 - working_dir, 151
- codar::cheetah::parameters::CodeCommand
 - __init__, 57
 - add_arg, 57
 - add_option, 57
 - args, 58
 - get_argv, 58
 - options, 58
 - target, 58
- codar::cheetah::parameters::Instance
 - __init__, 60
 - add_parameter, 60
 - as_dict, 60
 - as_string, 60
 - code_commands, 61
 - get_codes_argv, 61
 - get_hostfile, 61
 - get_nprocs, 61
 - get_parameter_values_by_type, 62
 - get_sched_opts, 62
 - parameter_values, 62
- codar::cheetah::parameters::Param
 - __get__, 91
 - __init__, 91
 - __len__, 92
 - name, 92
 - target, 92
 - values, 92
- codar::cheetah::parameters::ParamADIOS2XML
 - __init__, 94
 - io_name, 94
 - operation_name, 95
 - rc, 95
 - values, 95
- codar::cheetah::parameters::ParamAdiosXML
 - __init__, 97
 - group_name, 97
 - param_type, 97
 - var_name, 98
- codar::cheetah::parameters::ParamCmdLineArg
 - __init__, 99
 - position, 100
- codar::cheetah::parameters::ParamCmdLineOption
 - __init__, 101
 - option, 102
- codar::cheetah::parameters::ParamConfig
 - __init__, 103
 - config_filename, 104
 - match_string, 104
- codar::cheetah::parameters::ParamEnvVar
 - __init__, 105
 - option, 106
- codar::cheetah::parameters::ParamKeyValue
 - __init__, 109
 - config_filename, 110
 - key_name, 110
- codar::cheetah::parameters::ParamRunner
 - __init__, 111
- codar::cheetah::parameters::ParamSchedulerArgs
 - __init__, 113
- codar::cheetah::parameters::ParameterValue
 - __getattr__, 107
 - __init__, 107
 - is_type, 107
 - value, 108
- codar::cheetah::parameters::SummitOpts
 - __init__, 162
- codar::cheetah::parameters::Sweep
 - __init__, 167
 - get_instances, 167
 - node_layout, 168
 - parameters, 168
 - rc_dependency, 168
- codar::cheetah::parameters::SweepGroup
 - __init__, 170
 - component_inputs, 170
 - component_subdirs, 170
 - launch_mode, 170
 - max_procs, 170
 - name, 171
 - nodes, 171
 - parameter_groups, 171
 - per_run_timeout, 171
 - run_repetitions, 171
 - sosflow_analysis, 171
 - sosflow_profiling, 172
 - walltime, 172
- codar::cheetah::parameters::SymLink
 - __init__, 173
 - source, 173
- codar::cheetah::pbs
 - open_pbs_file, 23
 - PBS_FORMAT_TEMPLATE, 24
 - PBS_NAME, 24
 - SUBMIT_FORMAT_TEMPLATE, 24
 - write_run_script, 23
- codar::cheetah::report_generator
 - generate_report, 25

codar::cheetah::report_generator::_ReportGenerator
 __init__, 39
 campaign_directory, 41
 current_campaign_user, 41
 output_filename, 41
 parse_campaign, 40
 parse_run_dir, 40
 parse_sweep_group, 40
 parse_user_campaigns, 40
 parsed_runs, 41
 run_status, 42
 unique_keys, 42
 user_run_script, 42
 write_output, 41
 codar::cheetah::report_generator::_RunParser
 __init__, 43
 execute_user_run_script, 43
 exit_status, 46
 fob_dict, 46
 get_cheetah_perf_data, 43
 get_rc_names, 44
 get_run_params, 44
 rc_name_exe, 46
 rc_names, 46
 rc_working_dir, 46
 read_adios_output_file_sizes, 44
 read_fob_json, 44
 read_node_layout, 44
 read_sos_perf_data, 45
 run_dir, 46
 serialize_params_nested_dict, 45
 serialized_run_params, 47
 user_run_script, 47
 verify_run_successful, 45
 codar::cheetah::runners::Runner
 name, 154
 wrap_app_command, 154
 codar::cheetah::runners::RunnerCray
 name, 156
 wrap_app_command, 156
 codar::cheetah::runners::RunnerLocal
 name, 158
 wrap_app_command, 158
 codar::cheetah::status
 get_workflow_status, 26
 print_campaign_status, 26
 codar::cheetah::templates
 CAMPAIGN_ENV_TEMPLATE, 27
 GROUP_ENV_TEMPLATE, 27
 codar::savanna::consumer::PipelineRunner
 __init__, 123
 add_pipeline, 123
 allocated_nodes, 125
 free_cv, 125
 free_nodes, 126
 job_list, 126
 job_list_cv, 126
 kill_all, 124
 machine_name, 126
 max_nodes, 126
 pipeline_fatal, 124
 pipeline_finished, 124
 pipelines, 126
 pipelines_lock, 127
 ppn, 127
 run_finished, 124
 run_pipelines, 125
 runner, 127
 stop, 125
 codar::savanna::exc::MachineNotFound
 __init__, 79
 codar::savanna::machines
 cori, 29
 get_by_name, 29
 local, 30
 SCHEDULER_OPTIONS, 30
 summit, 30
 theta, 30
 titan, 31
 codar::savanna::machines::Machine
 __init__, 73
 dataspaces_servers_per_node, 74
 get_nodes_reqd, 74
 get_scheduler_options, 74
 name, 74
 node_class, 75
 node_exclusive, 75
 processes_per_node, 75
 runner_name, 75
 scheduler_name, 75
 scheduler_options, 75
 codar::savanna::machines::MachineNode
 __init__, 76
 cpu, 77
 gpu, 77
 to_json, 77
 validate_layout, 77
 codar::savanna::machines::SummitNode
 __init__, 160
 to_json, 161
 validate_layout, 161
 codar::savanna::main
 consumer, 32
 get_job_id, 31
 main, 32
 parse_args, 32
 codar::savanna::model
 KILL_WAIT, 33
 RETURN_NAME, 33
 STDERR_NAME, 33
 STDOUT_NAME, 33
 WAIT_DELAY_GIVE_UP, 33
 WAIT_DELAY_KILL, 34
 WALLTIME_NAME, 34
 codar::savanna::model::NodeConfig
 __init__, 84

- cpu, [84](#)
- gpu, [84](#)
- num_ranks_per_node, [84](#)
- codar::savanna::model::Pipeline
 - __init__, [115](#)
 - add_done_callback, [115](#)
 - add_fatal_callback, [115](#)
 - done_callbacks, [118](#)
 - fatal_callbacks, [119](#)
 - force_kill_all, [115](#)
 - from_data, [116](#)
 - get_nodes_used, [116](#)
 - get_pids, [116](#)
 - get_state, [116](#)
 - id, [119](#)
 - join_all, [117](#)
 - kill_on_partial_failure, [119](#)
 - launch_mode, [119](#)
 - log_prefix, [119](#)
 - machine_name, [119](#)
 - node_layout, [120](#)
 - nodes_assigned, [120](#)
 - post_process_args, [120](#)
 - post_process_script, [120](#)
 - post_process_stop_on_failure, [120](#)
 - remove_done_callback, [117](#)
 - remove_fatal_callback, [117](#)
 - run_finished, [117](#)
 - run_post_process_script, [117](#)
 - runs, [120](#)
 - set_ppn, [118](#)
 - set_total_nodes, [118](#)
 - start, [118](#)
 - total_nodes, [121](#)
 - total_procs, [121](#)
 - working_dir, [121](#)
- codar::savanna::model::Run
 - __init__, [132](#)
 - add_callback, [132](#)
 - args, [136](#)
 - callbacks, [136](#)
 - close, [132](#)
 - create_node_config, [133](#)
 - depends_on_runs, [136](#)
 - env, [136](#)
 - erf_file, [137](#)
 - exception, [133](#)
 - exe, [137](#)
 - from_data, [133](#)
 - get_nodes_used, [133](#)
 - get_pid, [134](#)
 - get_returncode, [134](#)
 - hostfile, [137](#)
 - join, [134](#)
 - kill, [134](#)
 - killed, [134](#)
 - log_prefix, [137](#)
 - machine, [137](#)
 - mpmd_run, [135](#)
 - name, [137](#)
 - node_config, [138](#)
 - nodes, [138](#)
 - nodes_assigned, [138](#)
 - nprocs, [138](#)
 - remove_callback, [135](#)
 - res_set, [138](#)
 - return_path, [138](#)
 - run, [135](#)
 - runner, [139](#)
 - runner_override, [139](#)
 - sched_args, [139](#)
 - set_runner, [135](#)
 - sleep_after, [139](#)
 - stderr_path, [139](#)
 - stdout_path, [139](#)
 - succeeded, [135](#)
 - tasks_per_node, [140](#)
 - timed_out, [136](#)
 - timeout, [140](#)
 - walltime_path, [140](#)
 - working_dir, [140](#)
- codar::savanna::node_layout::NodeLayout
 - __init__, [86](#)
 - add_node, [86](#)
 - as_data_list, [86](#)
 - codes_per_node, [87](#)
 - copy, [87](#)
 - default_no_share_layout, [87](#)
 - get_node_containing_code, [87](#)
 - group_codes_by_node, [87](#)
 - layout_list, [89](#)
 - layout_map, [89](#)
 - populate_remaining, [88](#)
 - ppn, [88](#)
 - serialize_to_dict, [88](#)
 - shared_nodes, [88](#)
 - validate, [89](#)
- codar::savanna::producer::JSONFilePipelineReader
 - __init__, [66](#)
 - file_path, [66](#)
 - read_pipelines, [66](#)
- codar::savanna::runners
 - aprun, [35](#)
 - jsrun, [35](#)
 - mpiexec, [35](#)
 - srun, [35](#)
- codar::savanna::runners::MPIRunner
 - __init__, [82](#)
 - exe, [82](#)
 - hostfile, [82](#)
 - nodes_arg, [83](#)
 - nprocs_arg, [83](#)
 - tasks_per_node_arg, [83](#)
 - wrap, [82](#)
- codar::savanna::runners::Runner
 - wrap, [152](#)

- codar::savanna::runners::SummitRunner
 - __init__, 163
 - bind_arg, 164
 - cpus_per_rs_arg, 164
 - exe, 165
 - gpus_per_rs_arg, 165
 - launch_distribution_arg, 165
 - machine, 165
 - nrs_arg, 165
 - rs_per_host_arg, 165
 - tasks_per_rs_arg, 166
 - wrap, 164
 - wrap_deprecated, 164
- codar::savanna::scheduler::JobList
 - __init__, 64
 - __len__, 64
 - add_job, 64
 - pop_job, 64
- codar::savanna::status
 - DONE, 36
 - KILLED, 36
 - NOT_STARTED, 37
 - REASON_EXCEPTION, 37
 - REASON_FAILED, 37
 - REASON_NOFIT, 37
 - REASON_SUCCEEDED, 37
 - REASON_TIMEOUT, 37
 - RUNNING, 38
- codar::savanna::status::PipelineState
 - __init__, 129
 - as_data, 129
 - id, 129
 - reason, 129
 - return_codes, 129
 - state, 130
- codar::savanna::status::WorkflowStatus
 - __init__, 175
 - file_path, 175
 - set_state, 175
- codar::savanna::summit_helper
 - create_erf_file, 38
 - get_nodes_reqd, 38
- code_commands
 - codar::cheetah::parameters::Instance, 61
- codes
 - codar::cheetah::model::Campaign, 50
 - codar::cheetah::model::Run, 144
- codes_path
 - codar::cheetah::model::Run, 144
- codes_per_node
 - codar::savanna::node_layout::NodeLayout, 87
- component_inputs
 - codar::cheetah::model::Run, 144
 - codar::cheetah::model::RunComponent, 149
 - codar::cheetah::parameters::SweepGroup, 170
- component_subdirs
 - codar::cheetah::model::Run, 144
 - codar::cheetah::parameters::SweepGroup, 170
- config_filename
 - codar::cheetah::parameters::ParamConfig, 104
 - codar::cheetah::parameters::ParamKeyValue, 110
- consumer
 - codar::savanna::main, 32
- copy
 - codar::savanna::node_layout::NodeLayout, 87
- copy_to_dir
 - codar::cheetah::helpers, 16
- copy_to_path
 - codar::cheetah::helpers, 16
- copytree_to_dir
 - codar::cheetah::helpers, 16
- cori
 - codar::savanna::machines, 29
- cpu
 - codar::savanna::machines::MachineNode, 77
 - codar::savanna::model::NodeConfig, 84
- cpus_per_rs_arg
 - codar::savanna::runners::SummitRunner, 164
- create_erf_file
 - codar::savanna::summit_helper, 38
- create_group_directory
 - codar::cheetah::launchers::Launcher, 68
- create_node_config
 - codar::savanna::model::Run, 133
- current_campaign_user
 - codar::cheetah::report_generator::_ReportGenerator, 41
- DATA_PATH
 - codar::cheetah::config, 14
- DONE
 - codar::savanna::status, 36
- dataspaces_servers_per_node
 - codar::savanna::machines::Machine, 74
- default_no_share_layout
 - codar::savanna::node_layout::NodeLayout, 87
- depends_on_runs
 - codar::savanna::model::Run, 136
- dir_size
 - codar::cheetah::helpers, 16
- done_callbacks
 - codar::savanna::model::Pipeline, 118
- env
 - codar::cheetah::model::RunComponent, 149
 - codar::savanna::model::Run, 136
- erf_file
 - codar::savanna::model::Run, 137
- etc_path
 - codar::cheetah::config, 13
- exception
 - codar::savanna::model::Run, 133
- exe
 - codar::cheetah::model::RunComponent, 149
 - codar::savanna::model::Run, 137
 - codar::savanna::runners::MPIRunner, 82
 - codar::savanna::runners::SummitRunner, 165

execute_user_run_script
 codar::cheetah::report_generator::_RunParser, 43
 exit_status
 codar::cheetah::report_generator::_RunParser, 46
 fatal_callbacks
 codar::savanna::model::Pipeline, 119
 file_path
 codar::savanna::producer::JSONFilePipeline←
 Reader, 66
 codar::savanna::status::WorkflowStatus, 175
 fob_dict
 codar::cheetah::report_generator::_RunParser, 46
 force_kill_all
 codar::savanna::model::Pipeline, 115
 free_cv
 codar::savanna::consumer::PipelineRunner, 125
 free_nodes
 codar::savanna::consumer::PipelineRunner, 126
 from_data
 codar::savanna::model::Pipeline, 116
 codar::savanna::model::Run, 133
 GROUP_ENV_TEMPLATE
 codar::cheetah::templates, 27
 generate_report
 codar::cheetah::report_generator, 25
 get_adios_version
 codar::cheetah::adios2_interface, 10
 get_app_param_dict
 codar::cheetah::model::Run, 143
 get_argv
 codar::cheetah::parameters::CodeCommand, 58
 get_by_name
 codar::savanna::machines, 29
 get_cheetah_perf_data
 codar::cheetah::report_generator::_RunParser, 43
 get_codes_argv
 codar::cheetah::parameters::Instance, 61
 get_dataspaces_num_servers
 codar::cheetah::config, 13
 get_file_size
 codar::cheetah::helpers, 17
 get_fob_data_list
 codar::cheetah::model::Run, 143
 get_hostfile
 codar::cheetah::parameters::Instance, 61
 get_immediate_subdirs
 codar::cheetah::helpers, 17
 get_instances
 codar::cheetah::parameters::Sweep, 167
 get_job_id
 codar::savanna::main, 31
 get_launcher
 codar::cheetah::machine_launchers, 21
 get_node_containing_code
 codar::savanna::node_layout::NodeLayout, 87
 get_nodes_reqd
 codar::savanna::machines::Machine, 74
 codar::savanna::summit_helper, 38
 get_nodes_used
 codar::savanna::model::Pipeline, 116
 codar::savanna::model::Run, 133
 get_nprocs
 codar::cheetah::parameters::Instance, 61
 get_parameter_values_by_type
 codar::cheetah::parameters::Instance, 62
 get_pid
 codar::savanna::model::Run, 134
 get_pids
 codar::savanna::model::Pipeline, 116
 get_rc_names
 codar::cheetah::report_generator::_RunParser, 44
 get_returncode
 codar::savanna::model::Run, 134
 get_run_params
 codar::cheetah::report_generator::_RunParser, 44
 get_sched_opts
 codar::cheetah::parameters::Instance, 62
 get_scheduler_options
 codar::savanna::machines::Machine, 74
 get_state
 codar::savanna::model::Pipeline, 116
 get_total_nprocs
 codar::cheetah::model::Run, 143
 get_workflow_status
 codar::cheetah::status, 26
 gpu
 codar::savanna::machines::MachineNode, 77
 codar::savanna::model::NodeConfig, 84
 gpus_per_rs_arg
 codar::savanna::runners::SummitRunner, 165
 group_codes_by_node
 codar::savanna::node_layout::NodeLayout, 87
 group_name
 codar::cheetah::parameters::ParamAdiosXML, 97
 hostfile
 codar::cheetah::model::RunComponent, 150
 codar::savanna::model::Run, 137
 codar::savanna::runners::MPIRunner, 82
 id
 codar::savanna::model::Pipeline, 119
 codar::savanna::status::PipelineState, 129
 inputs
 codar::cheetah::model::Campaign, 50
 codar::cheetah::model::Run, 144
 insert_sosflow
 codar::cheetah::model::Run, 143
 instance
 codar::cheetah::model::Run, 145
 io_name
 codar::cheetah::parameters::ParamADIOS2XML,
 94
 is_campaign_directory
 codar::cheetah::helpers, 17
 is_executable

- codar::cheetah::helpers, 17
- is_type
 - codar::cheetah::parameters::ParameterValue, 107
- job_list
 - codar::savanna::consumer::PipelineRunner, 126
- job_list_cv
 - codar::savanna::consumer::PipelineRunner, 126
- jobid_file_name
 - codar::cheetah::launchers::Launcher, 70
- join
 - codar::savanna::model::Run, 134
- join_all
 - codar::savanna::model::Pipeline, 117
- json_config_set_option
 - codar::cheetah::helpers, 18
- jsrun
 - codar::savanna::runners, 35
- KILL_WAIT
 - codar::savanna::model, 33
- KILLED
 - codar::savanna::status, 36
- key_name
 - codar::cheetah::parameters::ParamKeyValue, 110
- kill
 - codar::savanna::model::Run, 134
- kill_all
 - codar::savanna::consumer::PipelineRunner, 124
- kill_on_partial_failure
 - codar::cheetah::model::Campaign, 50
 - codar::savanna::model::Pipeline, 119
- killed
 - codar::savanna::model::Run, 134
- launch_distribution_arg
 - codar::savanna::runners::SummitRunner, 165
- launch_mode
 - codar::cheetah::parameters::SweepGroup, 170
 - codar::savanna::model::Pipeline, 119
- layout_list
 - codar::savanna::node_layout::NodeLayout, 89
- layout_map
 - codar::savanna::node_layout::NodeLayout, 89
- linked_with_sosflow
 - codar::cheetah::model::RunComponent, 150
- load_experiment_class
 - codar::cheetah::loader, 20
- local
 - codar::savanna::machines, 30
- log_prefix
 - codar::savanna::model::Pipeline, 119
 - codar::savanna::model::Run, 137
- machine
 - codar::cheetah::model::Campaign, 51
 - codar::cheetah::model::Run, 145
 - codar::savanna::model::Run, 137
 - codar::savanna::runners::SummitRunner, 165
- machine_app_config_script
 - codar::cheetah::model::Campaign, 51
- machine_launchers
 - codar::cheetah::machine_launchers, 21
- machine_name
 - codar::cheetah::launchers::Launcher, 70
 - codar::savanna::consumer::PipelineRunner, 126
 - codar::savanna::model::Pipeline, 119
- machine_scheduler_options
 - codar::cheetah::model::Campaign, 51
- machine_submit_env_path
 - codar::cheetah::config, 13
- main
 - codar::savanna::main, 32
- make_executable
 - codar::cheetah::helpers, 18
- make_experiment_run_dir
 - codar::cheetah::model::Campaign, 49
- match_string
 - codar::cheetah::parameters::ParamConfig, 104
- max_nodes
 - codar::savanna::consumer::PipelineRunner, 126
- max_procs
 - codar::cheetah::parameters::SweepGroup, 170
- mpiexec
 - codar::savanna::runners, 35
- mpmd_run
 - codar::savanna::model::Run, 135
- NOT_STARTED
 - codar::savanna::status, 37
- name
 - codar::cheetah::launchers::Launcher, 70
 - codar::cheetah::model::Campaign, 51
 - codar::cheetah::model::RunComponent, 150
 - codar::cheetah::parameters::Param, 92
 - codar::cheetah::parameters::SweepGroup, 171
 - codar::cheetah::runners::Runner, 154
 - codar::cheetah::runners::RunnerCray, 156
 - codar::cheetah::runners::RunnerLocal, 158
 - codar::savanna::machines::Machine, 74
 - codar::savanna::model::Run, 137
- node_class
 - codar::savanna::machines::Machine, 75
- node_config
 - codar::savanna::model::Run, 138
- node_exclusive
 - codar::savanna::machines::Machine, 75
- node_layout
 - codar::cheetah::model::Run, 145
 - codar::cheetah::parameters::Sweep, 168
 - codar::savanna::model::Pipeline, 120
- nodes
 - codar::cheetah::parameters::SweepGroup, 171
 - codar::savanna::model::Run, 138
- nodes_arg
 - codar::savanna::runners::MPIRunner, 83
- nodes_assigned
 - codar::savanna::model::Pipeline, 120

- codar::savanna::model::Run, 138
- nprocs
 - codar::cheetah::model::RunComponent, 150
 - codar::savanna::model::Run, 138
- nprocs_arg
 - codar::savanna::runners::MPIRunner, 83
- nrs_arg
 - codar::savanna::runners::SummitRunner, 165
- num_codes
 - codar::cheetah::launchers::Launcher, 70
- num_ranks_per_node
 - codar::savanna::model::NodeConfig, 84
- open_pbs_file
 - codar::cheetah::pbs, 23
- operation_name
 - codar::cheetah::parameters::ParamADIOS2XML, 95
- option
 - codar::cheetah::parameters::ParamCmdLine↵ Option, 102
 - codar::cheetah::parameters::ParamEnvVar, 106
- options
 - codar::cheetah::parameters::CodeCommand, 58
- output_directory
 - codar::cheetah::launchers::Launcher, 70
- output_filename
 - codar::cheetah::report_generator::_Report↵ Generator, 41
- PACKAGE_PATH
 - codar::cheetah::config, 15
- PBS_FORMAT_TEMPLATE
 - codar::cheetah::pbs, 24
- PBS_NAME
 - codar::cheetah::pbs, 24
- param_type
 - codar::cheetah::parameters::ParamAdiosXML, 97
- parameter_groups
 - codar::cheetah::parameters::SweepGroup, 171
- parameter_values
 - codar::cheetah::parameters::Instance, 62
- parameters
 - codar::cheetah::parameters::Sweep, 168
- parse_args
 - codar::savanna::main, 32
- parse_campaign
 - codar::cheetah::report_generator::_Report↵ Generator, 40
- parse_run_dir
 - codar::cheetah::report_generator::_Report↵ Generator, 40
- parse_sweep_group
 - codar::cheetah::report_generator::_Report↵ Generator, 40
- parse_timedelta_seconds
 - codar::cheetah::helpers, 18
- parse_user_campaigns
 - codar::cheetah::report_generator::_Report↵ Generator, 40
- parsed_runs
 - codar::cheetah::report_generator::_Report↵ Generator, 41
- per_run_timeout
 - codar::cheetah::parameters::SweepGroup, 171
- pipeline_fatal
 - codar::savanna::consumer::PipelineRunner, 124
- pipeline_finished
 - codar::savanna::consumer::PipelineRunner, 124
- pipelines
 - codar::savanna::consumer::PipelineRunner, 126
- pipelines_lock
 - codar::savanna::consumer::PipelineRunner, 127
- pop_job
 - codar::savanna::scheduler::JobList, 64
- populate_remaining
 - codar::savanna::node_layout::NodeLayout, 88
- position
 - codar::cheetah::parameters::ParamCmdLineArg, 100
- post_process_args
 - codar::savanna::model::Pipeline, 120
- post_process_script
 - codar::cheetah::model::Campaign, 51
 - codar::savanna::model::Pipeline, 120
- post_process_stop_on_failure
 - codar::savanna::model::Pipeline, 120
- ppn
 - codar::savanna::consumer::PipelineRunner, 127
 - codar::savanna::node_layout::NodeLayout, 88
- print_campaign_status
 - codar::cheetah::status, 26
- processes_per_node
 - codar::savanna::machines::Machine, 75
- python_path
 - codar::cheetah::model::Campaign, 51
- REASON_EXCEPTION
 - codar::savanna::status, 37
- REASON_FAILED
 - codar::savanna::status, 37
- REASON_NOFIT
 - codar::savanna::status, 37
- REASON_SUCCEEDED
 - codar::savanna::status, 37
- REASON_TIMEOUT
 - codar::savanna::status, 37
- RESERVED_CODE_NAMES
 - codar::cheetah::model, 22
- RETURN_NAME
 - codar::savanna::model, 33
- RUNNING
 - codar::savanna::status, 38
- rc
 - codar::cheetah::parameters::ParamADIOS2XML, 95
- rc_dependency

- codar::cheetah::parameters::Sweep, 168
- rc_name_exe
 - codar::cheetah::report_generator::_RunParser, 46
- rc_names
 - codar::cheetah::report_generator::_RunParser, 46
- rc_working_dir
 - codar::cheetah::report_generator::_RunParser, 46
- read_adios_output_file_sizes
 - codar::cheetah::report_generator::_RunParser, 44
- read_fob_json
 - codar::cheetah::report_generator::_RunParser, 44
- read_jobid
 - codar::cheetah::launchers::Launcher, 69
- read_node_layout
 - codar::cheetah::report_generator::_RunParser, 44
- read_pipelines
 - codar::savanna::producer::JSONFilePipeline←Reader, 66
- read_sos_perf_data
 - codar::cheetah::report_generator::_RunParser, 45
- reason
 - codar::savanna::status::PipelineState, 129
- relative_or_absolute_path
 - codar::cheetah::helpers, 18
- relative_or_absolute_path_list
 - codar::cheetah::helpers, 19
- remove_callback
 - codar::savanna::model::Run, 135
- remove_done_callback
 - codar::savanna::model::Pipeline, 117
- remove_fatal_callback
 - codar::savanna::model::Pipeline, 117
- require_campaign_directory
 - codar::cheetah::helpers, 19
- res_set
 - codar::savanna::model::Run, 138
- return_codes
 - codar::savanna::status::PipelineState, 129
- return_path
 - codar::savanna::model::Run, 138
- rs_per_host_arg
 - codar::savanna::runners::SummitRunner, 165
- run
 - codar::savanna::model::Run, 135
- run_command_name
 - codar::cheetah::launchers::Launcher, 70
- run_components
 - codar::cheetah::model::Run, 145
- run_dir
 - codar::cheetah::report_generator::_RunParser, 46
- run_dir_setup_script
 - codar::cheetah::model::Campaign, 52
- run_finished
 - codar::savanna::consumer::PipelineRunner, 124
 - codar::savanna::model::Pipeline, 117
- run_id
 - codar::cheetah::model::Run, 145
- run_json_name
 - codar::cheetah::launchers::Launcher, 71
- run_out_name
 - codar::cheetah::launchers::Launcher, 71
- run_path
 - codar::cheetah::model::Run, 145
- run_pipelines
 - codar::savanna::consumer::PipelineRunner, 125
- run_post_process_script
 - codar::cheetah::model::Campaign, 52
 - codar::savanna::model::Pipeline, 117
- run_post_process_stop_group_on_failure
 - codar::cheetah::model::Campaign, 52
- run_repetitions
 - codar::cheetah::parameters::SweepGroup, 171
- run_status
 - codar::cheetah::report_generator::_Report←Generator, 42
- runner
 - codar::savanna::consumer::PipelineRunner, 127
 - codar::savanna::model::Run, 139
- runner_name
 - codar::cheetah::launchers::Launcher, 71
 - codar::savanna::machines::Machine, 75
- runner_override
 - codar::cheetah::model::RunComponent, 150
 - codar::savanna::model::Run, 139
- runs
 - codar::cheetah::model::Campaign, 52
 - codar::savanna::model::Pipeline, 120
- SCHEDULER_OPTIONS
 - codar::savanna::machines, 30
- STDERR_NAME
 - codar::savanna::model, 33
- STDOUT_NAME
 - codar::savanna::model, 33
- SUBMIT_FORMAT_TEMPLATE
 - codar::cheetah::pbs, 24
- savanna/___init___py, 177
- savanna/consumer.py, 184
- savanna/exc.py, 179
- savanna/machines.py, 184
- savanna/main.py, 185
- savanna/model.py, 181
- savanna/node_layout.py, 185
- savanna/producer.py, 186
- savanna/runners.py, 183
- savanna/scheduler.py, 186
- savanna/status.py, 183
- savanna/submit_helper.py, 186
- sched_args
 - codar::cheetah::model::RunComponent, 150
 - codar::savanna::model::Run, 139
- scheduler_name
 - codar::cheetah::launchers::Launcher, 71
 - codar::savanna::machines::Machine, 75
- scheduler_options
 - codar::cheetah::model::Campaign, 52
 - codar::savanna::machines::Machine, 75

- scheduler_path
 - codar::cheetah::config, 14
- serialize_params_nested_dict
 - codar::cheetah::report_generator::_RunParser, 45
- serialize_to_dict
 - codar::savanna::node_layout::NodeLayout, 88
- serialized_run_params
 - codar::cheetah::report_generator::_RunParser, 47
- set_engine
 - codar::cheetah::adios2_interface, 10
- set_ppn
 - codar::savanna::model::Pipeline, 118
- set_runner
 - codar::savanna::model::Run, 135
- set_state
 - codar::savanna::status::WorkflowStatus, 175
- set_total_nodes
 - codar::savanna::model::Pipeline, 118
- set_transport
 - codar::cheetah::adios2_interface, 10
- set_var_operation
 - codar::cheetah::adios2_interface, 11
- shared_nodes
 - codar::savanna::node_layout::NodeLayout, 88
- sleep_after
 - codar::cheetah::model::RunComponent, 151
 - codar::savanna::model::Run, 139
- sos_analysis_path
 - codar::cheetah::model::Campaign, 52
- sosd_num_aggregators
 - codar::cheetah::model::Campaign, 53
- sosd_path
 - codar::cheetah::model::Campaign, 53
- sosflow_analysis
 - codar::cheetah::model::Run, 146
 - codar::cheetah::parameters::SweepGroup, 171
- sosflow_profiling
 - codar::cheetah::model::Run, 146
 - codar::cheetah::parameters::SweepGroup, 172
- source
 - codar::cheetah::parameters::SymLink, 173
- srun
 - codar::savanna::runners, 35
- start
 - codar::savanna::model::Pipeline, 118
- state
 - codar::savanna::status::PipelineState, 130
- status_script_name
 - codar::cheetah::launchers::Launcher, 71
- stderr_path
 - codar::savanna::model::Run, 139
- stdout_path
 - codar::savanna::model::Run, 139
- stop
 - codar::savanna::consumer::PipelineRunner, 125
- submit_out_name
 - codar::cheetah::launchers::Launcher, 71
- submit_script_name
 - codar::cheetah::launchers::Launcher, 72
- succeeded
 - codar::savanna::model::Run, 135
- summit
 - codar::savanna::machines, 30
- supported_machines
 - codar::cheetah::model::Campaign, 53
- sweeps
 - codar::cheetah::model::Campaign, 53
- swift_escape_string
 - codar::cheetah::helpers, 19
- TAU_PROFILE_PATTERN
 - codar::cheetah::launchers, 20
- target
 - codar::cheetah::parameters::CodeCommand, 58
 - codar::cheetah::parameters::Param, 92
- tasks_per_node
 - codar::savanna::model::Run, 140
- tasks_per_node_arg
 - codar::savanna::runners::MPIRunner, 83
- tasks_per_rs_arg
 - codar::savanna::runners::SummitRunner, 166
- tau_config
 - codar::cheetah::model::Campaign, 53
- theta
 - codar::savanna::machines, 30
- timed_out
 - codar::savanna::model::Run, 136
- timeout
 - codar::cheetah::model::RunComponent, 151
 - codar::savanna::model::Run, 140
- titan
 - codar::savanna::machines, 31
- to_json
 - codar::savanna::machines::MachineNode, 77
 - codar::savanna::machines::SummitNode, 161
- total_nodes
 - codar::cheetah::model::Run, 146
 - codar::savanna::model::Pipeline, 121
- total_procs
 - codar::savanna::model::Pipeline, 121
- umask
 - codar::cheetah::model::Campaign, 53
- unique_keys
 - codar::cheetah::report_generator::_Report↵Generator, 42
- user_run_script
 - codar::cheetah::report_generator::_Report↵Generator, 42
 - codar::cheetah::report_generator::_RunParser, 47
- validate
 - codar::savanna::node_layout::NodeLayout, 89
- validate_layout
 - codar::savanna::machines::MachineNode, 77
 - codar::savanna::machines::SummitNode, 161
- value

- codar::cheetah::parameters::ParameterValue, 108
- values
 - codar::cheetah::parameters::Param, 92
 - codar::cheetah::parameters::ParamADIOS2XML, 95
- var_name
 - codar::cheetah::parameters::ParamAdiosXML, 98
- verify_run_successful
 - codar::cheetah::report_generator::_RunParser, 45
- WAIT_DELAY_GIVE_UP
 - codar::savanna::model, 33
- WAIT_DELAY_KILL
 - codar::savanna::model, 34
- WALLTIME_NAME
 - codar::savanna::model, 34
- WORKFLOW_SCRIPT
 - codar::cheetah::config, 15
- wait_script_name
 - codar::cheetah::launchers::Launcher, 72
- walltime
 - codar::cheetah::parameters::SweepGroup, 172
- walltime_path
 - codar::savanna::model::Run, 140
- working_dir
 - codar::cheetah::model::RunComponent, 151
 - codar::savanna::model::Pipeline, 121
 - codar::savanna::model::Run, 140
- wrap
 - codar::savanna::runners::MPIRunner, 82
 - codar::savanna::runners::Runner, 152
 - codar::savanna::runners::SummitRunner, 164
- wrap_app_command
 - codar::cheetah::runners::Runner, 154
 - codar::cheetah::runners::RunnerCray, 156
 - codar::cheetah::runners::RunnerLocal, 158
- wrap_deprecated
 - codar::savanna::runners::SummitRunner, 164
- write_output
 - codar::cheetah::report_generator::_ReportGenerator, 41
- write_run_script
 - codar::cheetah::pbs, 23
- xml_has_transport
 - codar::cheetah::adios_params, 12