

ReferenceFrameModel

5.0

Generated by Doxygen 1.8.14

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Namespace List	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Data Structure Index	7
4.1	Data Structures	7
5	File Index	9
5.1	File List	9
6	Module Documentation	11
6.1	Models	11
6.1.1	Detailed Description	11
6.2	Utils	12
6.2.1	Detailed Description	12
6.3	RefFrames	13
6.3.1	Detailed Description	14
7	Namespace Documentation	15
7.1	jeod Namespace Reference	15
7.1.1	Detailed Description	16

8 Data Structure Documentation	17
8.1 jeod::ActivateInterface Class Reference	17
8.1.1 Detailed Description	17
8.1.2 Constructor & Destructor Documentation	17
8.1.2.1 ActivateInterface()	18
8.1.2.2 ~ActivateInterface()	18
8.1.3 Member Function Documentation	18
8.1.3.1 activate()	18
8.1.3.2 deactivate()	18
8.2 jeod::BaseRefFrameManager Class Reference	19
8.2.1 Detailed Description	20
8.2.2 Constructor & Destructor Documentation	20
8.2.2.1 ~BaseRefFrameManager()	20
8.2.3 Member Function Documentation	20
8.2.3.1 add_frame_to_tree()	20
8.2.3.2 add_ref_frame()	20
8.2.3.3 check_ref_frame_ownership()	21
8.2.3.4 find_ref_frame() [1/2]	21
8.2.3.5 find_ref_frame() [2/2]	21
8.2.3.6 frame_is_subscribed() [1/2]	22
8.2.3.7 frame_is_subscribed() [2/2]	22
8.2.3.8 remove_ref_frame()	22
8.2.3.9 reset_tree_root_node()	23
8.2.3.10 subscribe_to_frame() [1/2]	23
8.2.3.11 subscribe_to_frame() [2/2]	23
8.2.3.12 unsubscribe_to_frame() [1/2]	24
8.2.3.13 unsubscribe_to_frame() [2/2]	24
8.2.4 Friends And Related Function Documentation	24
8.2.4.1 init_attrjeod__BaseRefFrameManager	24
8.2.4.2 InputProcessor	24

8.3	jeod::JeodLinksIterators< Links > Struct Template Reference	25
8.3.1	Detailed Description	25
8.3.2	Member Typedef Documentation	25
8.3.2.1	ForwardIterator	25
8.3.2.2	ReverselIterator	25
8.4	jeod::JeodLinksIterators< const Links > Struct Template Reference	26
8.4.1	Detailed Description	26
8.4.2	Member Typedef Documentation	26
8.4.2.1	ForwardIterator	26
8.4.2.2	ReverselIterator	26
8.5	jeod::RefFrame Class Reference	27
8.5.1	Detailed Description	29
8.5.2	Constructor & Destructor Documentation	30
8.5.2.1	RefFrame() [1/2]	30
8.5.2.2	RefFrame() [2/2]	30
8.5.2.3	~RefFrame()	30
8.5.3	Member Function Documentation	30
8.5.3.1	add_child()	30
8.5.3.2	compute_position_from()	31
8.5.3.3	compute_pred_rel_state() [1/2]	31
8.5.3.4	compute_pred_rel_state() [2/2]	32
8.5.3.5	compute_relative_state() [1/2]	32
8.5.3.6	compute_relative_state() [2/2]	33
8.5.3.7	compute_state_wrt_pred() [1/2]	34
8.5.3.8	compute_state_wrt_pred() [2/2]	34
8.5.3.9	find_last_common_index()	35
8.5.3.10	find_last_common_node()	35
8.5.3.11	get_name()	36
8.5.3.12	get_owner()	36
8.5.3.13	get_parent()	37

8.5.3.14	get_root()	37
8.5.3.15	is_progeny_of()	37
8.5.3.16	make_root()	38
8.5.3.17	operator=()	38
8.5.3.18	remove_from_parent()	38
8.5.3.19	reset_parent()	38
8.5.3.20	set_active_status()	39
8.5.3.21	set_name() [1/7]	39
8.5.3.22	set_name() [2/7]	40
8.5.3.23	set_name() [3/7]	40
8.5.3.24	set_name() [4/7]	40
8.5.3.25	set_name() [5/7]	41
8.5.3.26	set_name() [6/7]	41
8.5.3.27	set_name() [7/7]	42
8.5.3.28	set_owner()	42
8.5.3.29	set_timestamp()	43
8.5.3.30	timestamp()	43
8.5.3.31	transplant_node()	43
8.5.4	Friends And Related Function Documentation	44
8.5.4.1	init_attrjeod__RefFrame	44
8.5.4.2	InputProcessor	44
8.5.4.3	RefFrameLinks	44
8.5.5	Field Documentation	44
8.5.5.1	links	44
8.5.5.2	name	45
8.5.5.3	owner	45
8.5.5.4	state	45
8.5.5.5	update_time	45
8.6	jeod::RefFrameItems Class Reference	46
8.6.1	Detailed Description	47

8.6.2	Member Enumeration Documentation	47
8.6.2.1	Items	47
8.6.3	Constructor & Destructor Documentation	48
8.6.3.1	RefFrameItems() [1/2]	48
8.6.3.2	RefFrameItems() [2/2]	48
8.6.4	Member Function Documentation	48
8.6.4.1	add()	48
8.6.4.2	contains()	49
8.6.4.3	equals()	49
8.6.4.4	get()	50
8.6.4.5	is_empty()	50
8.6.4.6	is_full()	50
8.6.4.7	remove()	50
8.6.4.8	set()	51
8.6.4.9	to_string() [1/2]	51
8.6.4.10	to_string() [2/2]	52
8.6.5	Friends And Related Function Documentation	52
8.6.5.1	init_attrjeod__RefFrameItems	52
8.6.5.2	InputProcessor	52
8.6.6	Field Documentation	52
8.6.6.1	value	53
8.7	jeod::RefFrameLinks Class Reference	53
8.7.1	Detailed Description	54
8.7.2	Constructor & Destructor Documentation	54
8.7.2.1	RefFrameLinks() [1/3]	54
8.7.2.2	~RefFrameLinks()	54
8.7.2.3	RefFrameLinks() [2/3]	55
8.7.2.4	RefFrameLinks() [3/3]	55
8.7.3	Member Function Documentation	55
8.7.3.1	operator=()	55

8.7.4	Friends And Related Function Documentation	55
8.7.4.1	init_attrjeod__RefFrameLinks	55
8.7.4.2	InputProcessor	56
8.7.5	Field Documentation	56
8.7.5.1	default_path_size	56
8.8	jeod::RefFrameManager Class Reference	56
8.8.1	Detailed Description	58
8.8.2	Constructor & Destructor Documentation	58
8.8.2.1	RefFrameManager() [1/2]	58
8.8.2.2	~RefFrameManager()	58
8.8.2.3	RefFrameManager() [2/2]	58
8.8.3	Member Function Documentation	58
8.8.3.1	add_frame_to_tree()	58
8.8.3.2	add_ref_frame()	59
8.8.3.3	check_ref_frame_ownership()	59
8.8.3.4	find_ref_frame() [1/2]	59
8.8.3.5	find_ref_frame() [2/2]	60
8.8.3.6	frame_is_subscribed() [1/2]	60
8.8.3.7	frame_is_subscribed() [2/2]	61
8.8.3.8	operator=()	61
8.8.3.9	remove_ref_frame()	61
8.8.3.10	reset_tree_root_node()	62
8.8.3.11	subscribe_to_frame() [1/2]	62
8.8.3.12	subscribe_to_frame() [2/2]	63
8.8.3.13	unsubscribe_to_frame() [1/2]	63
8.8.3.14	unsubscribe_to_frame() [2/2]	64
8.8.3.15	validate_name()	64
8.8.4	Friends And Related Function Documentation	65
8.8.4.1	init_attrjeod__RefFrameManager	65
8.8.4.2	InputProcessor	65

8.8.5	Field Documentation	65
8.8.5.1	ref_frames	65
8.8.5.2	root_node	65
8.9	jeod::RefFrameMessages Class Reference	66
8.9.1	Detailed Description	67
8.9.2	Constructor & Destructor Documentation	67
8.9.2.1	RefFrameMessages() [1/2]	67
8.9.2.2	RefFrameMessages() [2/2]	67
8.9.3	Member Function Documentation	67
8.9.3.1	operator=()	67
8.9.4	Friends And Related Function Documentation	67
8.9.4.1	init_attrjeod__RefFrameMessages	67
8.9.4.2	InputProcessor	68
8.9.5	Field Documentation	68
8.9.5.1	attach_info	68
8.9.5.2	duplicate_entry	68
8.9.5.3	inconsistent_setup	68
8.9.5.4	internal_error	69
8.9.5.5	invalid_attach	69
8.9.5.6	invalid_detach	69
8.9.5.7	invalid_enum	69
8.9.5.8	invalid_item	70
8.9.5.9	invalid_name	70
8.9.5.10	invalid_node	70
8.9.5.11	null_pointer	70
8.9.5.12	removal_failed	71
8.9.5.13	subscription_error	71
8.10	jeod::RefFrameOwner Class Reference	71
8.10.1	Detailed Description	72
8.10.2	Constructor & Destructor Documentation	72

8.10.2.1	RefFrameOwner()	72
8.10.2.2	~RefFrameOwner()	72
8.10.3	Member Function Documentation	72
8.10.3.1	note_frame_status_change()	72
8.11	jeod::RefFrameRot Class Reference	73
8.11.1	Detailed Description	74
8.11.2	Constructor & Destructor Documentation	74
8.11.2.1	RefFrameRot() [1/2]	74
8.11.2.2	RefFrameRot() [2/2]	74
8.11.2.3	~RefFrameRot()	74
8.11.3	Member Function Documentation	75
8.11.3.1	compute_ang_vel_products()	75
8.11.3.2	compute_ang_vel_unit()	75
8.11.3.3	compute_quaternion()	75
8.11.3.4	compute_transformation()	76
8.11.3.5	copy()	76
8.11.3.6	initialize()	76
8.11.3.7	operator=()	76
8.11.4	Friends And Related Function Documentation	77
8.11.4.1	init_attrjeod__RefFrameRot	77
8.11.4.2	InputProcessor	77
8.11.5	Field Documentation	77
8.11.5.1	ang_vel_mag	77
8.11.5.2	ang_vel_this	78
8.11.5.3	ang_vel_unit	78
8.11.5.4	Q_parent_this	78
8.11.5.5	T_parent_this	79
8.12	jeod::RefFrameState Class Reference	79
8.12.1	Detailed Description	80
8.12.2	Constructor & Destructor Documentation	80

8.12.2.1	RefFrameState() [1/2]	80
8.12.2.2	RefFrameState() [2/2]	80
8.12.2.3	~RefFrameState()	81
8.12.3	Member Function Documentation	81
8.12.3.1	copy()	81
8.12.3.2	decr_left()	81
8.12.3.3	decr_right()	82
8.12.3.4	incr_left()	82
8.12.3.5	incr_right()	83
8.12.3.6	initialize()	83
8.12.3.7	negate()	83
8.12.3.8	operator=()	84
8.12.4	Friends And Related Function Documentation	84
8.12.4.1	init_attrjeod__RefFrameState	84
8.12.4.2	InputProcessor	84
8.12.5	Field Documentation	84
8.12.5.1	rot	85
8.12.5.2	trans	85
8.13	jeod::RefFrameTrans Class Reference	85
8.13.1	Detailed Description	86
8.13.2	Constructor & Destructor Documentation	86
8.13.2.1	RefFrameTrans() [1/2]	86
8.13.2.2	RefFrameTrans() [2/2]	86
8.13.2.3	~RefFrameTrans()	87
8.13.3	Member Function Documentation	87
8.13.3.1	copy()	87
8.13.3.2	initialize()	87
8.13.3.3	operator=()	88
8.13.4	Friends And Related Function Documentation	88
8.13.4.1	init_attrjeod__RefFrameTrans	88

8.13.4.2	InputProcessor	88
8.13.5	Field Documentation	88
8.13.5.1	position	88
8.13.5.2	velocity	89
8.14	jeod::SubscribeInterface Class Reference	89
8.14.1	Detailed Description	89
8.14.2	Constructor & Destructor Documentation	90
8.14.2.1	SubscribeInterface()	90
8.14.2.2	~SubscribeInterface()	90
8.14.3	Member Function Documentation	90
8.14.3.1	unsubscribe()	90
8.14.3.2	subscribe()	90
8.15	jeod::Subscription Class Reference	91
8.15.1	Detailed Description	92
8.15.2	Member Enumeration Documentation	92
8.15.2.1	Mode	92
8.15.3	Constructor & Destructor Documentation	93
8.15.3.1	Subscription() [1/2]	93
8.15.3.2	Subscription() [2/2]	93
8.15.3.3	~Subscription()	93
8.15.4	Member Function Documentation	93
8.15.4.1	activate()	94
8.15.4.2	deactivate()	94
8.15.4.3	get_subscription_mode()	94
8.15.4.4	is_active()	95
8.15.4.5	set_active_status()	95
8.15.4.6	set_subscription_mode()	95
8.15.4.7	subscribe()	96
8.15.4.8	subscriptions()	96
8.15.4.9	unsubscribe()	97

8.15.5 Friends And Related Function Documentation	97
8.15.5.1 init_attrjeod__Subscription	97
8.15.5.2 InputProcessor	97
8.15.6 Field Documentation	97
8.15.6.1 active	97
8.15.6.2 mode	98
8.15.6.3 subscribers	98
8.16 jeod::TreeLinks< Links, Container, Messages > Class Template Reference	98
8.16.1 Detailed Description	101
8.16.2 Constructor & Destructor Documentation	102
8.16.2.1 TreeLinks() [1/3]	102
8.16.2.2 ~TreeLinks()	102
8.16.2.3 TreeLinks() [2/3]	103
8.16.2.4 TreeLinks() [3/3]	103
8.16.3 Member Function Documentation	103
8.16.3.1 attach()	103
8.16.3.2 attach_internal()	103
8.16.3.3 child_head()	104
8.16.3.4 child_tail()	104
8.16.3.5 construct_path_to_node()	104
8.16.3.6 container() [1/2]	105
8.16.3.7 container() [2/2]	105
8.16.3.8 detach()	105
8.16.3.9 detach_internal()	105
8.16.3.10 find_last_common_index()	105
8.16.3.11 find_last_common_node()	106
8.16.3.12 find_path_index()	106
8.16.3.13 has_children()	107
8.16.3.14 is_atomic()	107
8.16.3.15 is_progeny_of()	107

8.16.3.16	is_root()	108
8.16.3.17	links_parent() [1/2]	108
8.16.3.18	links_parent() [2/2]	108
8.16.3.19	links_root() [1/2]	109
8.16.3.20	links_root() [2/2]	109
8.16.3.21	make_root()	109
8.16.3.22	nth_from_root() [1/2]	109
8.16.3.23	nth_from_root() [2/2]	110
8.16.3.24	operator=()	110
8.16.3.25	parent() [1/2]	110
8.16.3.26	parent() [2/2]	111
8.16.3.27	path_length()	111
8.16.3.28	reattach()	111
8.16.3.29	root() [1/2]	112
8.16.3.30	root() [2/2]	112
8.16.3.31	set_path_size()	112
8.16.4	Friends And Related Function Documentation	113
8.16.4.1	init_attrjeod__TreeLinks	113
8.16.4.2	InputProcessor	113
8.16.4.3	TreeLinksAscendRange	113
8.16.4.4	TreeLinksChildrenRange	113
8.16.4.5	TreeLinksDescentRange	114
8.16.5	Field Documentation	114
8.16.5.1	children_	114
8.16.5.2	container_	114
8.16.5.3	parent_	115
8.16.5.4	path_to_node_	115
8.17	jeod::TreeLinksAscendRange< Links > Class Template Reference	116
8.17.1	Detailed Description	116
8.17.2	Member Typedef Documentation	116

8.17.2.1	Reverseliterator	116
8.17.3	Constructor & Destructor Documentation	117
8.17.3.1	TreeLinksAscendRange() [1/2]	117
8.17.3.2	TreeLinksAscendRange() [2/2]	117
8.18	jeod::TreeLinksChildIterator< Links, Container > Class Template Reference	117
8.18.1	Detailed Description	118
8.19	jeod::TreeLinksChildrenRange< Links > Class Template Reference	118
8.19.1	Detailed Description	118
8.19.2	Member Typedef Documentation	118
8.19.2.1	ForwardIterator	119
8.19.3	Constructor & Destructor Documentation	119
8.19.3.1	TreeLinksChildrenRange()	119
8.20	jeod::TreeLinksDescentIterator< Links, Container > Class Template Reference	119
8.20.1	Detailed Description	119
8.21	jeod::TreeLinksDescentRange< Links > Class Template Reference	120
8.21.1	Detailed Description	120
8.21.2	Member Typedef Documentation	120
8.21.2.1	ForwardIterator	120
8.21.3	Constructor & Destructor Documentation	121
8.21.3.1	TreeLinksDescentRange()	121
8.22	jeod::TreeLinksIterator< Links, Container > Class Template Reference	121
8.22.1	Detailed Description	121
8.23	jeod::TreeLinksParentIterator< Links, Container > Class Template Reference	121
8.23.1	Detailed Description	122
8.24	jeod::TreeLinksRange< Iterator > Class Template Reference	122
8.24.1	Detailed Description	122
8.24.2	Constructor & Destructor Documentation	123
8.24.2.1	TreeLinksRange()	123
8.24.3	Member Function Documentation	123
8.24.3.1	begin()	123
8.24.3.2	end()	124
8.24.4	Field Documentation	124
8.24.4.1	begin_	124
8.24.4.2	end_	124

9	File Documentation	125
9.1	base_ref_frame_manager.hh File Reference	125
9.1.1	Detailed Description	125
9.2	class_declarations.hh File Reference	125
9.2.1	Detailed Description	126
9.3	ref_frame.cc File Reference	126
9.3.1	Detailed Description	126
9.4	ref_frame.hh File Reference	126
9.4.1	Detailed Description	127
9.5	ref_frame_compute_relative_state.cc File Reference	127
9.5.1	Detailed Description	127
9.6	ref_frame_inline.hh File Reference	127
9.6.1	Detailed Description	128
9.7	ref_frame_interface.hh File Reference	128
9.7.1	Detailed Description	128
9.8	ref_frame_items.cc File Reference	128
9.8.1	Detailed Description	128
9.9	ref_frame_items.hh File Reference	129
9.9.1	Detailed Description	129
9.10	ref_frame_items_inline.hh File Reference	129
9.10.1	Detailed Description	129
9.11	ref_frame_links.hh File Reference	129
9.11.1	Detailed Description	130
9.12	ref_frame_manager.cc File Reference	130
9.12.1	Detailed Description	130
9.13	ref_frame_manager.hh File Reference	130
9.13.1	Detailed Description	131
9.14	ref_frame_messages.cc File Reference	131
9.14.1	Detailed Description	131
9.14.2	Macro Definition Documentation	131

9.14.2.1	MAKE_REF_FRAME_MESSAGE_CODE	131
9.15	ref_frame_messages.hh File Reference	132
9.15.1	Detailed Description	132
9.16	ref_frame_set_name.cc File Reference	132
9.16.1	Detailed Description	132
9.17	ref_frame_state.cc File Reference	132
9.17.1	Detailed Description	133
9.18	ref_frame_state.hh File Reference	133
9.18.1	Detailed Description	133
9.19	ref_frame_state_inline.hh File Reference	133
9.19.1	Detailed Description	134
9.20	subscription.cc File Reference	134
9.20.1	Detailed Description	134
9.21	subscription.hh File Reference	134
9.21.1	Detailed Description	135
9.22	tree_links.hh File Reference	135
9.22.1	Detailed Description	135
9.23	tree_links_iterator.hh File Reference	136
9.23.1	Detailed Description	136
Index		137

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Models	11
Utils	12
RefFrames	13

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

jeod	Namespace jeod	15
----------------------	--------------------------	--------------------

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

jeod::ActivateInterface	17
jeod::BaseRefFrameManager	19
jeod::RefFrameManager	56
jeod::JeodLinksIterators< Links >	25
jeod::JeodLinksIterators< const Links >	26
jeod::RefFrameItems	46
jeod::RefFrameMessages	66
jeod::RefFrameOwner	71
jeod::RefFrameRot	73
jeod::RefFrameState	79
jeod::RefFrameTrans	85
jeod::SubscribeInterface	89
jeod::Subscription	91
jeod::RefFrame	27
jeod::TreeLinks< Links, Container, Messages >	98
jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >	98
jeod::RefFrameLinks	53
jeod::TreeLinksChildIterator< Links, Container >	117
jeod::TreeLinksDescentIterator< Links, Container >	119
jeod::TreeLinksIterator< Links, Container >	121
jeod::TreeLinksParentIterator< Links, Container >	121
jeod::TreeLinksRange< Iterator >	122
jeod::TreeLinksRange< JeodLinksIterators< Links >::ForwardIterator >	122
jeod::TreeLinksChildrenRange< Links >	118
jeod::TreeLinksDescentRange< Links >	120
jeod::TreeLinksRange< JeodLinksIterators< Links >::ReverseIterator >	122
jeod::TreeLinksAscendRange< Links >	116

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

jeod::ActivateInterface	A class that inherits from the ActivateInterface class must provide activate and deactivate methods	17
jeod::BaseRefFrameManager	The RefFrameManager class manages the reference frames in a simulation	19
jeod::JeodLinksIterators< Links >	Class template that defines member types ForwardIterator and ReverseIterator for walking over a std::vector of pointers to Links objects	25
jeod::JeodLinksIterators< const Links >	Partial specialization of JeodLinksIterators for const Links types	26
jeod::RefFrame	Describe a frame of reference and define operations on reference frames	27
jeod::RefFrameItems	Identify which aspects of a reference frame's state have been set	46
jeod::RefFrameLinks	Encapsulates the links between reference frames	53
jeod::RefFrameManager	Manages the reference frames in a simulation	56
jeod::RefFrameMessages	Declares messages associated with the reference frames model	66
jeod::RefFrameOwner	Identify an object as an "owner" of a reference frame	71
jeod::RefFrameRot	Represent the rotational aspects of a reference frame's state	73
jeod::RefFrameState	Represent a reference frame's state	79
jeod::RefFrameTrans	Represent the translational aspects of a reference frame's state	85
jeod::SubscribeInterface	A class that inherits from the SubscribeInterface class must provide subscribe and unsubscribe methods	89
jeod::Subscription	A Subscription object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods	91
jeod::TreeLinks< Links, Container, Messages >	Encapsulates links (parent, children, siblings) between objects, in the form of a tree	98

jeod::TreeLinksAscendRange< Links >	
A TreeLinksAscendRange walks up a Links object's path_to_node_data member, starting at the start node and ending just before the end node	116
jeod::TreeLinksChildIterator< Links, Container >	117
jeod::TreeLinksChildrenRange< Links >	
A TreeLinksChildrenRange walks over a Links object's children_	118
jeod::TreeLinksDescentIterator< Links, Container >	119
jeod::TreeLinksDescentRange< Links >	
A TreeLinksDescentRange walks down a Links object's path_to_node_data member, starting at the start node and ending just before the end node	120
jeod::TreeLinksIterator< Links, Container >	121
jeod::TreeLinksParentIterator< Links, Container >	121
jeod::TreeLinksRange< Iterator >	
Base class template for all tree links range types	122

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

base_ref_frame_manager.hh	Define the BaseRefFrameManager class, which defines the interfaces but not the implementations of the class RefFrameManager	125
class_declarations.hh	Forward declarations of classes defined in ref_frame.hh	125
ref_frame.cc	Define basic methods for the RefFrame class	126
ref_frame.hh	Define the class RefFrame	126
ref_frame_compute_relative_state.cc	Define relative state methods for the RefFrame class	127
ref_frame_inline.hh	Define inline methods for the RefFrame class	127
ref_frame_interface.hh	Define the class RefFrameOwner, which identifies an object as an "owner" of a reference frame	128
ref_frame_items.cc	Define basic methods for the RefFrameState class	128
ref_frame_items.hh	Define the class RefFrameItems, which identifies the aspects of a reference frame's state that have been set	129
ref_frame_items_inline.hh	Define inline functions for the RefFrameItems::Items	129
ref_frame_links.hh	Define the class RefFrameLinks, the class that encapsulates the links between reference frames	129
ref_frame_manager.cc	Define RefFrameManager methods	130
ref_frame_manager.hh	Define the RefFrameManager class, which manages the reference frames in a JEOD-based simulation	130
ref_frame_messages.cc	Implement the class RefFrameMessages	131
ref_frame_messages.hh	Define the class RefFrameMessages, the class that specifies the message IDs used in the reference frames model	132
ref_frame_set_name.cc	Define the RefFrame::set_name methods	132

ref_frame_state.cc	
Define methods for the RefFrameState class	132
ref_frame_state.hh	
JEOD 2.0 reference frame tree class definitions	133
ref_frame_state_inline.hh	
Define inline methods for the RefFrameState class and its component	133
subscription.cc	
Define non-inlined methods for the Subscription class	134
subscription.hh	
Define the class Subscription	134
tree_links.hh	
Define the template class TreeLinks, the class that encapsulates the parent/ child links between objects	135
tree_links_iterator.hh	
Define the template TreeLinksRange and related templates, which are used to iterate over trees	136

Chapter 6

Module Documentation

6.1 Models

Modules

- [Utils](#)

6.1.1 Detailed Description

6.2 Utils

Modules

- [RefFrames](#)

6.2.1 Detailed Description

6.3 RefFrames

Files

- file [base_ref_frame_manager.hh](#)
Define the BaseRefFrameManager class, which defines the interfaces but not the implementations of the class RefFrameManager.
- file [class_declarations.hh](#)
Forward declarations of classes defined in [ref_frame.hh](#).
- file [ref_frame.hh](#)
Define the class RefFrame.
- file [ref_frame_inline.hh](#)
Define inline methods for the RefFrame class.
- file [ref_frame_interface.hh](#)
Define the class RefFrameOwner, which identifies an object as an "owner" of a reference frame.
- file [ref_frame_items.hh](#)
Define the class RefFrameItems, which identifies the aspects of a reference frame's state that have been set.
- file [ref_frame_items_inline.hh](#)
Define inline functions for the RefFrameItems::Items.
- file [ref_frame_links.hh](#)
Define the class RefFrameLinks, the class that encapsulates the links between reference frames.
- file [ref_frame_manager.hh](#)
Define the RefFrameManager class, which manages the reference frames in a JEOD-based simulation.
- file [ref_frame_messages.hh](#)
Define the class RefFrameMessages, the class that specifies the message IDs used in the reference frames model.
- file [ref_frame_state.hh](#)
JEOD 2.0 reference frame tree class definitions.
- file [ref_frame_state_inline.hh](#)
Define inline methods for the RefFrameState class and its component.
- file [subscription.hh](#)
Define the class Subscription.
- file [tree_links.hh](#)
Define the template class TreeLinks, the class that encapsulates the parent/ child links between objects.
- file [tree_links_iterator.hh](#)
Define the template TreeLinksRange and related templates, which are used to iterate over trees.
- file [ref_frame.cc](#)
Define basic methods for the RefFrame class.
- file [ref_frame_compute_relative_state.cc](#)
Define relative state methods for the RefFrame class.
- file [ref_frame_items.cc](#)
Define basic methods for the RefFrameState class.
- file [ref_frame_manager.cc](#)
Define RefFrameManager methods.
- file [ref_frame_messages.cc](#)
Implement the class RefFrameMessages.
- file [ref_frame_set_name.cc](#)
Define the RefFrame::set_name methods.
- file [ref_frame_state.cc](#)
Define methods for the RefFrameState class.
- file [subscription.cc](#)
Define non-inlined methods for the Subscription class.

Namespaces

- [jeod](#)

Namespace jeod.

6.3.1 Detailed Description

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class [ActivateInterface](#)
A class that inherits from the [ActivateInterface](#) class must provide activate and deactivate methods.
- class [BaseRefFrameManager](#)
The [RefFrameManager](#) class manages the reference frames in a simulation.
- struct [JeodLinksIterators](#)
Class template that defines member types ForwardIterator and ReverselIterator for walking over a std::vector of pointers to Links objects.
- struct [JeodLinksIterators< const Links >](#)
Partial specialization of [JeodLinksIterators](#) for const Links types.
- class [RefFrame](#)
Describe a frame of reference and define operations on reference frames.
- class [RefFrameItems](#)
Identify which aspects of a reference frame's state have been set.
- class [RefFrameLinks](#)
Encapsulates the links between reference frames.
- class [RefFrameManager](#)
The [RefFrameManager](#) class manages the reference frames in a simulation.
- class [RefFrameMessages](#)
Declares messages associated with the reference frames model.
- class [RefFrameOwner](#)
Identify an object as an "owner" of a reference frame.
- class [RefFrameRot](#)
Represent the rotational aspects of a reference frame's state.
- class [RefFrameState](#)
Represent a reference frame's state.
- class [RefFrameTrans](#)
Represent the translational aspects of a reference frame's state.
- class [SubscribeInterface](#)

A class that inherits from the [SubscribeInterface](#) class must provide subscribe and unsubscribe methods.

- class [Subscription](#)

A [Subscription](#) object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods.

- class [TreeLinks](#)

Encapsulates links (parent, children, siblings) between objects, in the form of a tree.

- class [TreeLinksAscendRange](#)

A [TreeLinksAscendRange](#) walks up a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

- class [TreeLinksChildIterator](#)

- class [TreeLinksChildrenRange](#)

A [TreeLinksChildrenRange](#) walks over a Links object's children_.

- class [TreeLinksDescentIterator](#)

- class [TreeLinksDescentRange](#)

A [TreeLinksDescentRange](#) walks down a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

- class [TreeLinksIterator](#)

- class [TreeLinksParentIterator](#)

- class [TreeLinksRange](#)

Base class template for all tree links range types.

7.1.1 Detailed Description

Namespace jeod.

Chapter 8

Data Structure Documentation

8.1 jeod::ActivateInterface Class Reference

A class that inherits from the [ActivateInterface](#) class must provide activate and deactivate methods.

```
#include <subscription.hh>
```

Public Member Functions

- [ActivateInterface](#) ()

Default constructor.

- virtual [~ActivateInterface](#) ()

Destructor.

- virtual void [activate](#) (void)=0

Mark the object as active.

- virtual void [deactivate](#) (void)=0

Mark the object as inactive.

8.1.1 Detailed Description

A class that inherits from the [ActivateInterface](#) class must provide activate and deactivate methods.

Definition at line 79 of file subscription.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 ActivateInterface()

```
jeod::ActivateInterface::ActivateInterface ( ) [inline]
```

Default constructor.

Definition at line 89 of file subscription.hh.

8.1.2.2 ~ActivateInterface()

```
virtual jeod::ActivateInterface::~~ActivateInterface ( ) [inline], [virtual]
```

Destructor.

Definition at line 94 of file subscription.hh.

8.1.3 Member Function Documentation

8.1.3.1 activate()

```
virtual void jeod::ActivateInterface::activate (  
    void ) [pure virtual]
```

Mark the object as active.

8.1.3.2 deactivate()

```
virtual void jeod::ActivateInterface::deactivate (  
    void ) [pure virtual]
```

Mark the object as inactive.

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

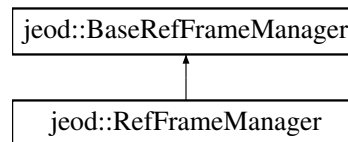
- [subscription.hh](#)

8.2 jeod::BaseRefFrameManager Class Reference

The [RefFrameManager](#) class manages the reference frames in a simulation.

```
#include <base_ref_frame_manager.hh>
```

Inheritance diagram for jeod::BaseRefFrameManager:



Public Member Functions

- virtual [~BaseRefFrameManager](#) ()
Destructor.
- virtual void [add_ref_frame](#) ([RefFrame](#) &ref_frame)=0
Add a reference frame to the list of such.
- virtual void [remove_ref_frame](#) ([RefFrame](#) &ref_frame)=0
Remove a reference frame from the list of such.
- virtual [RefFrame](#) * [find_ref_frame](#) (const char *name) const =0
Find a reference frame.
- virtual [RefFrame](#) * [find_ref_frame](#) (const char *prefix, const char *suffix) const =0
Find a reference frame.
- virtual void [check_ref_frame_ownership](#) () const =0
Check whether each reference frame has an owner.
- virtual void [reset_tree_root_node](#) ()=0
Reset the root node in anticipation of rebuilding the entire tree.
- virtual void [add_frame_to_tree](#) ([RefFrame](#) &ref_frame, [RefFrame](#) *parent)=0
Add a reference frame to the reference frame tree.
- virtual void [subscribe_to_frame](#) (const char *frame_name)=0
Add a subscription to a reference frame.
- virtual void [subscribe_to_frame](#) ([RefFrame](#) &frame)=0
Add a subscription to a reference frame.
- virtual void [unsubscribe_to_frame](#) (const char *frame_name)=0
Remove a subscription from a reference frame.
- virtual void [unsubscribe_to_frame](#) ([RefFrame](#) &frame)=0
Remove a subscription from a reference frame.
- virtual bool [frame_is_subscribed](#) (const char *frame_name)=0
Check whether a reference frame has subscriptions.
- virtual bool [frame_is_subscribed](#) ([RefFrame](#) &frame)=0
Check whether a reference frame has subscriptions.

Friends

- class [InputProcessor](#)
- void [init_attrjeod__BaseRefFrameManager](#) ()

8.2.1 Detailed Description

The [RefFrameManager](#) class manages the reference frames in a simulation.

This class defines the external interfaces to that class.

Definition at line 80 of file `base_ref_frame_manager.hh`.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 `~BaseRefFrameManager()`

```
virtual jeod::BaseRefFrameManager::~~BaseRefFrameManager ( ) [inline], [virtual]
```

Destructor.

Definition at line 93 of file `base_ref_frame_manager.hh`.

8.2.3 Member Function Documentation

8.2.3.1 `add_frame_to_tree()`

```
virtual void jeod::BaseRefFrameManager::add_frame_to_tree (
    RefFrame & ref_frame,
    RefFrame * parent ) [pure virtual]
```

Add a reference frame to the reference frame tree.

Parameters

<i>ref_frame</i>	Frame to be added.
<i>parent</i>	Parent of the frame.

Implemented in [jeod::RefFrameManager](#).

8.2.3.2 `add_ref_frame()`

```
virtual void jeod::BaseRefFrameManager::add_ref_frame (
    RefFrame & ref_frame ) [pure virtual]
```

Add a reference frame to the list of such.

Parameters

<i>ref_frame</i>	Frame to be added.
------------------	--------------------

Implemented in [jeod::RefFrameManager](#).

8.2.3.3 check_ref_frame_ownership()

```
virtual void jeod::BaseRefFrameManager::check_ref_frame_ownership ( ) const [pure virtual]
```

Check whether each reference frame has an owner.

Implemented in [jeod::RefFrameManager](#).

8.2.3.4 find_ref_frame() [1/2]

```
virtual RefFrame* jeod::BaseRefFrameManager::find_ref_frame (
    const char * name ) const [pure virtual]
```

Find a reference frame.

Parameters

<i>name</i>	Frame to be found.
-------------	--------------------

Returns

Found reference frame.

Implemented in [jeod::RefFrameManager](#).

8.2.3.5 find_ref_frame() [2/2]

```
virtual RefFrame* jeod::BaseRefFrameManager::find_ref_frame (
    const char * prefix,
    const char * suffix ) const [pure virtual]
```

Find a reference frame.

Parameters

<i>prefix</i>	Prefix of frame to be found.
<i>suffix</i>	Suffix of frame to be found.

Returns

Found reference frame.

Implemented in [jeod::RefFrameManager](#).

8.2.3.6 frame_is_subscribed() [1/2]

```
virtual bool jeod::BaseRefFrameManager::frame_is_subscribed (
    const char * frame_name ) [pure virtual]
```

Check whether a reference frame has subscriptions.

Parameters

<i>frame_name</i>	Frame to be checked.
-------------------	----------------------

Returns

True if frame has subscriptions, false otherwise.

Implemented in [jeod::RefFrameManager](#).

8.2.3.7 frame_is_subscribed() [2/2]

```
virtual bool jeod::BaseRefFrameManager::frame_is_subscribed (
    RefFrame & frame ) [pure virtual]
```

Check whether a reference frame has subscriptions.

Parameters

<i>frame</i>	Frame to be checked.
--------------	----------------------

Returns

True if frame has subscriptions, false otherwise.

Implemented in [jeod::RefFrameManager](#).

8.2.3.8 remove_ref_frame()

```
virtual void jeod::BaseRefFrameManager::remove_ref_frame (
    RefFrame & ref_frame ) [pure virtual]
```

Remove a reference frame from the list of such.

Parameters

<i>ref_frame</i>	Frame to be removed.
------------------	----------------------

Implemented in [jeod::RefFrameManager](#).

8.2.3.9 reset_tree_root_node()

```
virtual void jeod::BaseRefFrameManager::reset_tree_root_node ( ) [pure virtual]
```

Reset the root node in anticipation of rebuilding the entire tree.

Implemented in [jeod::RefFrameManager](#).

8.2.3.10 subscribe_to_frame() [1/2]

```
virtual void jeod::BaseRefFrameManager::subscribe_to_frame (
    const char * frame_name ) [pure virtual]
```

Add a subscription to a reference frame.

Parameters

<i>frame_name</i>	Frame to which subscription is to be issued.
-------------------	--

Implemented in [jeod::RefFrameManager](#).

8.2.3.11 subscribe_to_frame() [2/2]

```
virtual void jeod::BaseRefFrameManager::subscribe_to_frame (
    RefFrame & frame ) [pure virtual]
```

Add a subscription to a reference frame.

Parameters

<i>frame</i>	Frame to which subscription is to be issued.
--------------	--

Implemented in [jeod::RefFrameManager](#).

8.2.3.12 unsubscribe_to_frame() [1/2]

```
virtual void jeod::BaseRefFrameManager::unsubscribe_to_frame (
    const char * frame_name ) [pure virtual]
```

Remove a subscription from a reference frame.

Parameters

<i>frame_name</i>	Frame from which subscription is to be removed.
-------------------	---

Implemented in [jeod::RefFrameManager](#).

8.2.3.13 unsubscribe_to_frame() [2/2]

```
virtual void jeod::BaseRefFrameManager::unsubscribe_to_frame (
    RefFrame & frame ) [pure virtual]
```

Remove a subscription from a reference frame.

Parameters

<i>frame</i>	Frame from which subscription is to be removed.
--------------	---

Implemented in [jeod::RefFrameManager](#).

8.2.4 Friends And Related Function Documentation**8.2.4.1 init_attrjeod__BaseRefFrameManager**

```
void init_attrjeod__BaseRefFrameManager ( ) [friend]
```

8.2.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 82 of file `base_ref_frame_manager.hh`.

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

- [base_ref_frame_manager.hh](#)

8.3 jeod::JeodLinksIterators< Links > Struct Template Reference

Class template that defines member types ForwardIterator and ReverseIterator for walking over a std::vector of pointers to Links objects.

```
#include <tree_links_iterator.hh>
```

Public Types

- using [ForwardIterator](#) = typename std::vector< Links * >::iterator
- using [ReverseIterator](#) = typename std::vector< Links * >::reverse_iterator

8.3.1 Detailed Description

```
template<class Links>
struct jeod::JeodLinksIterators< Links >
```

Class template that defines member types ForwardIterator and ReverseIterator for walking over a std::vector of pointers to Links objects.

This primary template definition is for a non-const Links type.

Template Parameters

<i>Links</i>	Link object type.
--------------	-------------------

Definition at line 93 of file tree_links_iterator.hh.

8.3.2 Member Typedef Documentation

8.3.2.1 ForwardIterator

```
template<class Links>
using jeod::JeodLinksIterators< Links >::ForwardIterator = typename std::vector<Links*>↵
::iterator
```

Definition at line 95 of file tree_links_iterator.hh.

8.3.2.2 ReverseIterator

```
template<class Links>
using jeod::JeodLinksIterators< Links >::ReverseIterator = typename std::vector<Links*>↵
::reverse_iterator
```

Definition at line 96 of file tree_links_iterator.hh.

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

- [tree_links_iterator.hh](#)

8.4 jeod::JeodLinksIterators< const Links > Struct Template Reference

Partial specialization of [JeodLinksIterators](#) for const Links types.

```
#include <tree_links_iterator.hh>
```

Public Types

- using [ForwardIterator](#) = typename std::vector< Links * >::const_iterator
- using [ReverseIterator](#) = typename std::vector< Links * >::const_reverse_iterator

8.4.1 Detailed Description

```
template<class Links>
struct jeod::JeodLinksIterators< const Links >
```

Partial specialization of [JeodLinksIterators](#) for const Links types.

Like the primary definition, this specialization defines member types [ForwardIterator](#) and [ReverseIterator](#), but this are now const iterators.

Template Parameters

<i>Links</i>	Link object type.
--------------	-------------------

Definition at line 107 of file `tree_links_iterator.hh`.

8.4.2 Member Typedef Documentation

8.4.2.1 ForwardIterator

```
template<class Links >
using jeod::JeodLinksIterators< const Links >::ForwardIterator = typename std::vector<Links*>↵
::const_iterator
```

Definition at line 109 of file `tree_links_iterator.hh`.

8.4.2.2 ReverseIterator

```
template<class Links >
using jeod::JeodLinksIterators< const Links >::ReverseIterator = typename std::vector<Links*>↵
::const_reverse_iterator
```

Definition at line 110 of file `tree_links_iterator.hh`.

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

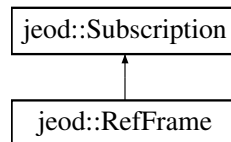
- [tree_links_iterator.hh](#)

8.5 jeod::RefFrame Class Reference

Describe a frame of reference and define operations on reference frames.

```
#include <ref_frame.hh>
```

Inheritance diagram for jeod::RefFrame:



Public Member Functions

- [RefFrame](#) (void)
Construct a [RefFrame](#) object.
- virtual [~RefFrame](#) (void)
Destroy a [RefFrame](#) object.
- void [set_name](#) (const char *name_item1)
Create a copy of the provided name.
- void [set_name](#) (const char *name_item1, const char *name_item2)
Construct a name as a dot-conjoined string.
- void [set_name](#) (const char *name_item1, const char *name_item2, const char *name_item3)
Construct a name as a dot-conjoined string.
- void [set_name](#) (const char *name_item1, const char *name_item2, const char *name_item3, const char *name_item4)
Construct a name as a dot-conjoined string.
- void [set_name](#) (const char *name_item1, const char *name_item2, const char *name_item3, const char *name_item4, const char *name_item5)
Construct a name as a dot-conjoined string.
- void [set_name](#) (const char *name_item1, const char *name_item2, const char *name_item3, const char *name_item4, const char *name_item5, const char *name_item6)
Construct a name as a dot-conjoined string.
- void [set_name](#) (const char *name_item1, const char *name_item2, const char *name_item3, const char *name_item4, const char *name_item5, const char *name_item6, const char *name_item7)
Construct a name as a dot-conjoined string.
- virtual const char * [get_name](#) (void) const
Return the name.
- virtual void [set_timestamp](#) (double time)
Set the update time of this frame.
- virtual double [timestamp](#) (void) const
Return the update time of this frame.
- virtual void [set_owner](#) (RefFrameOwner *new_owner)
Set the owner of this frame.
- virtual RefFrameOwner * [get_owner](#) (void) const
Return the owner of this frame.
- virtual void [set_active_status](#) (bool value)
Augment [Subscription::set_active_status](#) by telling the frame owner that the active/inactive state of this frame has changed.

- const [RefFrame](#) * [get_parent](#) (void) const
Return the parent of this frame.
- const [RefFrame](#) * [get_root](#) (void) const
Return the root of this frame's tree.
- virtual void [make_root](#) (void)
Make this frame a root frame.
- virtual void [add_child](#) ([RefFrame](#) &frame)
Add a child frame to this frame.
- virtual void [remove_from_parent](#) (void)
Remove this node as a child of its parent node.
- bool [is_progeny_of](#) (const [RefFrame](#) &frame) const
Return true if this frame is a progeny of the provided frame, false if not.
- virtual void [transplant_node](#) ([RefFrame](#) &new_parent)
Move a node to a different place in the tree, keeping the state with respect to the root frame constant.
- virtual void [reset_parent](#) ([RefFrame](#) &new_parent)
Reparent a node, without updating state.
- virtual void [compute_relative_state](#) (const [RefFrame](#) &wrt_frame, [RefFrameState](#) &rel_state) const
*Compute the complete state of the invoking reference frame (*this) with respect to the supplied wrt_frame reference frame.*
- virtual void [compute_relative_state](#) (const [RefFrame](#) &wrt_frame, bool reverse_sense, [RefFrameState](#) &rel_state) const
*Compute the complete state of the invoking reference frame (*this) with respect to the supplied wrt_frame reference frame.*
- virtual void [compute_state_wrt_pred](#) (const [RefFrame](#) &wrt_frame, [RefFrameState](#) &rel_state) const
Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which must be a predecessor of the invoking frame.
- virtual void [compute_state_wrt_pred](#) (unsigned int wrt_frame_index, [RefFrameState](#) &rel_state) const
Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which must be a predecessor of the invoking frame.
- virtual void [compute_pred_rel_state](#) (const [RefFrame](#) &wrt_frame, [RefFrameState](#) &rel_state) const
Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which must be a predecessor of the invoking frame.
- virtual void [compute_pred_rel_state](#) (unsigned int wrt_frame_index, [RefFrameState](#) &rel_state) const
Compute the complete state of the supplied reference frame wrt the invoking reference frame.
- virtual void [compute_position_from](#) (const [RefFrame](#) &in_frame, double rel_pos[3]) const
Compute the relative position vector from the origin of the supplied reference frame to the origin of this reference frame, expressed in the coordinates of the supplied frame.
- const [RefFrame](#) * [find_last_common_node](#) (const [RefFrame](#) &frame) const
Each reference frame has a path from the root of the reference frame tree to the frame in question.

Data Fields

- [RefFrameState](#) state
The translational and rotational state of the reference frame with respect to its parent.

Protected Member Functions

- int [find_last_common_index](#) (const [RefFrame](#) &frame) const
Each reference frame has a path from the root of the reference frame tree to the frame in question.

Protected Attributes

- `std::string name`
The identifier for this reference frame.
- `RefFrameOwner * owner`
The object that "owns" this frame.
- `RefFrameLinks links`
Specifies the parent/child/sibling linkages between frames.
- `double update_time`
The time that the frame was last updated, dynamic time seconds.

Private Member Functions

- `RefFrame (const RefFrame &frame)`
Not implemented.
- `RefFrame & operator= (const RefFrame &frame)`
Not implemented.

Friends

- class `InputProcessor`
- class `RefFrameLinks`
- void `init_attrjeod__RefFrame ()`

Additional Inherited Members

8.5.1 Detailed Description

Describe a frame of reference and define operations on reference frames.

A JEOD reference frame

- Is characterized by an origin and a set of three orthogonal axes.
- Provides a mechanism for specifying the translational and rotational states of an object in space (particularly, Cartesian three space).
- Is itself an object whose translational and rotational states can be specified/determined in terms of some other reference frame.
- Is a node in a rooted tree of reference frames, each of which has some specific state with respect to another node in the tree.
- Can be active (or inactive). An active frame supposedly will have a (fairly) current state. All bets are off if the frame is inactive.
- Can have subscribers, which are external entities that for some reason need the frame to be active.

Reference frames are one of the key concepts that define JEOD 2.0.

Definition at line 99 of file `ref_frame.hh`.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 RefFrame() [1/2]

```
jeod::RefFrame::RefFrame (
    const RefFrame & frame ) [private]
```

Not implemented.

8.5.2.2 RefFrame() [2/2]

```
jeod::RefFrame::RefFrame (
    void )
```

Construct a [RefFrame](#) object.

Definition at line 49 of file `ref_frame.cc`.

References `jeod::Subscription::set_subscription_mode()`, and `jeod::Subscription::Subscribe`.

8.5.2.3 ~RefFrame()

```
jeod::RefFrame::~~RefFrame (
    void ) [virtual]
```

Destroy a [RefFrame](#) object.

Definition at line 64 of file `ref_frame.cc`.

References `jeod::TreeLinks< Links, Container, Messages >::child_tail()`, `jeod::TreeLinks< Links, Container, Messages >::detach()`, `jeod::TreeLinks< Links, Container, Messages >::has_children()`, `links`, and `remove_from_parent()`.

8.5.3 Member Function Documentation

8.5.3.1 add_child()

```
void jeod::RefFrame::add_child (
    RefFrame & frame ) [inline], [virtual]
```

Add a child frame to this frame.

Parameters

<i>in</i> , <i>out</i>	<i>frame</i>	Frame to add as child
------------------------	--------------	-----------------------

Definition at line 190 of file `ref_frame_inline.hh`.

References `jeod::TreeLinks< Links, Container, Messages >::attach()`, and `links`.

Referenced by `jeod::RefFrameManager::add_frame_to_tree()`.

8.5.3.2 compute_position_from()

```
void jeod::RefFrame::compute_position_from (
    const RefFrame & in_frame,
    double rel_pos[3] ) const [virtual]
```

Compute the relative position vector from the origin of the supplied reference frame to the origin of this reference frame, expressed in the coordinates of the supplied frame.

Parameters

<i>in</i>	<i>in_frame</i>	Relative position vector origin
<i>out</i>	<i>rel_pos</i>	Relative position vector Units: M

Definition at line 354 of file `ref_frame_compute_relative_state.cc`.

References `find_last_common_index()`, `jeod::RefFrameMessages::invalid_node`, `links`, `name`, `jeod::TreeLinks< Links, Container, Messages >::path_length()`, `jeod::RefFrameTrans::position`, `jeod::RefFrameRot::Q_parent_this`, `jeod::RefFrameState::rot`, `state`, `jeod::RefFrameRot::T_parent_this`, and `jeod::RefFrameState::trans`.

8.5.3.3 compute_pred_rel_state() [1/2]

```
void jeod::RefFrame::compute_pred_rel_state (
    const RefFrame & pred_frame,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which *must* be a predecessor of the invoking frame.

Assumptions and Limitations

- The predecessor frame is a predecessor.

Parameters

in	<i>pred_frame</i>	The frame with respect to which the state is to be expressed
out	<i>rel_state</i>	The relative state

Definition at line 279 of file `ref_frame_compute_relative_state.cc`.

References `jeod::TreeLinks< Links, Container, Messages >::find_path_index()`, `jeod::RefFrameMessages<::invalid_node, links, name, and jeod::TreeLinks< Links, Container, Messages >::path_length()`.

Referenced by `compute_relative_state()`.

8.5.3.4 compute_pred_rel_state() [2/2]

```
void jeod::RefFrame::compute_pred_rel_state (
    unsigned int pred_frame_index,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the supplied reference frame wrt the invoking reference frame.

The supplied reference frame must be a predecessor of the invoking frame.

Assumptions and Limitations

- The predecessor frame is a predecessor.

Parameters

in	<i>pred_frame_index</i>	The frame with respect to which the state is to be expressed
out	<i>rel_state</i>	The relative state

Definition at line 313 of file `ref_frame_compute_relative_state.cc`.

References `jeod::RefFrameState::decr_right()`, `links`, `jeod::RefFrameState::negate()`, `jeod::TreeLinks< Links, Container, Messages >::path_length()`, and `state`.

8.5.3.5 compute_relative_state() [1/2]

```
void jeod::RefFrame::compute_relative_state (
    const RefFrame & wrt_frame,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the invoking reference frame (`*this`) with respect to the supplied `wrt_frame` reference frame.

The state will include:

- The position and velocity of the invoking frame with respect to the supplied `wrt_frame`, expressed in the coordinates of the `wrt_frame`.
- The angular velocity of the invoking frame with respect to the supplied `wrt_frame`, expressed in the coordinates of invoking frame.
- The transformation (as a matrix and a quaternion) from the supplied `wrt_frame` to the invoking frame.

Assumptions and Limitations

- The two frames are in the same tree.

Parameters

in	<i>wrt_frame</i>	The frame with respect to which the state is to be expressed
out	<i>rel_state</i>	The relative state

Definition at line 63 of file `ref_frame_compute_relative_state.cc`.

References `compute_pred_rel_state()`, `compute_state_wrt_pred()`, `jeod::RefFrameState::decr_left()`, `find_last_common_index()`, `jeod::RefFrameState::initialize()`, `jeod::RefFrameMessages::invalid_node`, `links`, `name`, and `jeod::TreeLinks< Links, Container, Messages >::nth_from_root()`.

Referenced by `compute_relative_state()`, and `transplant_node()`.

8.5.3.6 compute_relative_state() [2/2]

```
void jeod::RefFrame::compute_relative_state (
    const RefFrame & wrt_frame,
    bool reverse_sense,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the invoking reference frame (*this) with respect to the supplied `wrt_frame` reference frame.

If `reverse_sense` is false, the results are those from the simpler two argument form of [RefFrame::compute_relative_state](#). If `reverse_sense` is true, the results from the two argument form are transformed as follows:

- The position and velocity are those the invoking frame with respect to the supplied `wrt_frame`, but expressed in invoking frame coordinates.
- The angular velocity of the invoking frame with respect to the supplied `wrt_frame`, expressed in the coordinates of supplied `wrt_frame`.
- The transformation (as a matrix and a quaternion) from the invoking frame to the supplied `wrt_frame`.

Assumptions and Limitations

- The two frames are in the same tree.

Parameters

in	<i>wrt_frame</i>	The frame with respect to which the state is to be expressed
in	<i>reverse_sense</i>	Express position and velocity in this frame, angular velocity in the wrt_frame, and the transformations from this frame to the wrt_frame.
out	<i>rel_state</i>	The relative state

Definition at line 162 of file `ref_frame_compute_relative_state.cc`.

References `jeod::RefFrameRot::ang_vel_this`, `jeod::RefFrameRot::ang_vel_unit`, `compute_relative_state()`, `jeod::RefFrameTrans::position`, `jeod::RefFrameRot::Q_parent_this`, `jeod::RefFrameState::rot`, `jeod::RefFrameRot::T_parent_this`, `jeod::RefFrameState::trans`, and `jeod::RefFrameTrans::velocity`.

8.5.3.7 compute_state_wrt_pred() [1/2]

```
void jeod::RefFrame::compute_state_wrt_pred (
    const RefFrame & pred_frame,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which *must* be a predecessor of the invoking frame.

Assumptions and Limitations

- The predecessor frame is a predecessor.

Parameters

in	<i>pred_frame</i>	The frame with respect to which the state is to be expressed
out	<i>rel_state</i>	The relative state

Definition at line 203 of file `ref_frame_compute_relative_state.cc`.

References `jeod::TreeLinks< Links, Container, Messages >::find_path_index()`, `jeod::RefFrameMessages::invalid_node`, `links`, `name`, and `jeod::TreeLinks< Links, Container, Messages >::path_length()`.

Referenced by `compute_relative_state()`.

8.5.3.8 compute_state_wrt_pred() [2/2]

```
void jeod::RefFrame::compute_state_wrt_pred (
    unsigned int pred_frame_index,
    RefFrameState & rel_state ) const [virtual]
```

Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which *must* be a predecessor of the invoking frame.

Assumptions and Limitations

- The predecessor frame is a predecessor.

Parameters

in	<i>pred_frame_index</i>	The frame with respect to which the state is to be expressed
out	<i>rel_state</i>	The relative state

Definition at line 237 of file `ref_frame_compute_relative_state.cc`.

References `jeod::RefFrameState::copy()`, `jeod::RefFrameState::incr_left()`, `links`, `jeod::TreeLinks< Links, Container, Messages >::path_length()`, and `state`.

8.5.3.9 find_last_common_index()

```
int jeod::RefFrame::find_last_common_index (
    const RefFrame & frame ) const [inline], [protected]
```

Each reference frame has a path from the root of the reference frame tree to the frame in question.

The paths for two reference frames will have some initial sequence of common nodes. Find the index number of this last element in this sequence.

Returns

Last common node

Parameters

in	<i>frame</i>	Other frame
----	--------------	-------------

Definition at line 221 of file `ref_frame_inline.hh`.

References `jeod::TreeLinks< Links, Container, Messages >::find_last_common_index()`, and `links`.

Referenced by `compute_position_from()`, and `compute_relative_state()`.

8.5.3.10 find_last_common_node()

```
const RefFrame * jeod::RefFrame::find_last_common_node (
    const RefFrame & frame ) const [inline]
```

Each reference frame has a path from the root of the reference frame tree to the frame in question.

The paths for two reference frames will have some initial sequence of common nodes. Find the last element in this sequence.

Returns

Last common node

Parameters

in	<i>frame</i>	Other frame
----	--------------	-------------

Definition at line 238 of file `ref_frame_inline.hh`.

References `jeod::TreeLinks< Links, Container, Messages >::container()`, `jeod::TreeLinks< Links, Container, Messages >::find_last_common_node()`, and `links`.

8.5.3.11 get_name()

```
const char * jeod::RefFrame::get_name (
    void ) const [inline], [virtual]
```

Return the name.

Returns

Void

Definition at line 86 of file `ref_frame_inline.hh`.

References `name`.

Referenced by `jeod::RefFrameManager::add_ref_frame()`, `jeod::RefFrameManager::check_ref_frame_ownership()`, `jeod::RefFrameManager::find_ref_frame()`, `jeod::RefFrameManager::remove_ref_frame()`, and `jeod::RefFrameManager::unsubscribe_to_frame()`.

8.5.3.12 get_owner()

```
RefFrameOwner * jeod::RefFrame::get_owner (
    void ) const [inline], [virtual]
```

Return the owner of this frame.

Returns

Frame owner

Definition at line 112 of file `ref_frame_inline.hh`.

References `owner`.

Referenced by `jeod::RefFrameManager::check_ref_frame_ownership()`.

8.5.3.13 get_parent()

```
const RefFrame * jeod::RefFrame::get_parent (
    void ) const [inline]
```

Return the parent of this frame.

Returns

Frame parent

Definition at line 125 of file ref_frame_inline.hh.

References links, and jeod::TreeLinks< Links, Container, Messages >::parent().

8.5.3.14 get_root()

```
const RefFrame * jeod::RefFrame::get_root (
    void ) const [inline]
```

Return the root of this frame's tree.

Returns

Tree root

Definition at line 138 of file ref_frame_inline.hh.

References links, and jeod::TreeLinks< Links, Container, Messages >::root().

8.5.3.15 is_progeny_of()

```
bool jeod::RefFrame::is_progeny_of (
    const RefFrame & frame ) const [inline]
```

Return true if this frame is a progeny of the provided frame, false if not.

Returns

This is progeny of frame

Parameters

in	<i>frame</i>	Other frame
----	--------------	-------------

Definition at line 260 of file ref_frame_inline.hh.

References `jeod::TreeLinks< Links, Container, Messages >::is_progeny_of()`, and `links`.

8.5.3.16 `make_root()`

```
void jeod::RefFrame::make_root (
    void ) [inline], [virtual]
```

Make this frame a root frame.

Definition at line 176 of file `ref_frame_inline.hh`.

References `links`, and `jeod::TreeLinks< Links, Container, Messages >::make_root()`.

Referenced by `jeod::RefFrameManager::add_frame_to_tree()`.

8.5.3.17 `operator=()`

```
RefFrame& jeod::RefFrame::operator= (
    const RefFrame & frame ) [private]
```

Not implemented.

8.5.3.18 `remove_from_parent()`

```
void jeod::RefFrame::remove_from_parent (
    void ) [inline], [virtual]
```

Remove this node as a child of its parent node.

Definition at line 203 of file `ref_frame_inline.hh`.

References `jeod::TreeLinks< Links, Container, Messages >::detach()`, and `links`.

Referenced by `~RefFrame()`.

8.5.3.19 `reset_parent()`

```
void jeod::RefFrame::reset_parent (
    RefFrame & new_parent ) [virtual]
```

Reparent a node, without updating state.

Parameters

in	<i>new_parent</i>	New parent frame
----	-------------------	------------------

Definition at line 123 of file `ref_frame.cc`.

References `links`, and `jeod::TreeLinks< Links, Container, Messages >::reattach()`.

8.5.3.20 `set_active_status()`

```
void jeod::RefFrame::set_active_status (
    bool value ) [virtual]
```

Augment [Subscription::set_active_status](#) by telling the frame owner that the active/inactive state of this frame has changed.

Parameters

in	<i>value</i>	New active value
----	--------------	------------------

Reimplemented from [jeod::Subscription](#).

Definition at line 82 of file `ref_frame.cc`.

References `jeod::RefFrameOwner::note_frame_status_change()`, `owner`, and `jeod::Subscription::set_active_status()`.

8.5.3.21 `set_name()` [1/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1 )
```

Create a copy of the provided name.

Parameters

in	<i>name_item1</i>	First part of the name
----	-------------------	------------------------

Definition at line 42 of file `ref_frame_set_name.cc`.

References `name`.

Referenced by `set_name()`.

8.5.3.22 set_name() [2/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name

Definition at line 55 of file `ref_frame_set_name.cc`.

References `name`.

8.5.3.23 set_name() [3/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2,
    const char * name_item3 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name
in	<i>name_item3</i>	Third part of the name

Definition at line 71 of file `ref_frame_set_name.cc`.

References `name`, and `set_name()`.

8.5.3.24 set_name() [4/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2,
    const char * name_item3,
    const char * name_item4 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name
in	<i>name_item3</i>	Third part of the name
in	<i>name_item4</i>	Fourth part of the name

Definition at line 92 of file `ref_frame_set_name.cc`.

References `name`, and `set_name()`.

8.5.3.25 `set_name()` [5/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2,
    const char * name_item3,
    const char * name_item4,
    const char * name_item5 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name
in	<i>name_item3</i>	Third part of the name
in	<i>name_item4</i>	Fourth part of the name
in	<i>name_item5</i>	Fifth part of the name

Definition at line 115 of file `ref_frame_set_name.cc`.

References `name`, and `set_name()`.

8.5.3.26 `set_name()` [6/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2,
    const char * name_item3,
    const char * name_item4,
    const char * name_item5,
    const char * name_item6 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name
in	<i>name_item3</i>	Third part of the name
in	<i>name_item4</i>	Fourth part of the name
in	<i>name_item5</i>	Fifth part of the name
in	<i>name_item6</i>	Sixth part of the name

Definition at line 140 of file `ref_frame_set_name.cc`.

References `name`, and `set_name()`.

8.5.3.27 set_name() [7/7]

```
void jeod::RefFrame::set_name (
    const char * name_item1,
    const char * name_item2,
    const char * name_item3,
    const char * name_item4,
    const char * name_item5,
    const char * name_item6,
    const char * name_item7 )
```

Construct a name as a dot-conjoined string.

Parameters

in	<i>name_item1</i>	First part of the name
in	<i>name_item2</i>	Second part of the name
in	<i>name_item3</i>	Third part of the name
in	<i>name_item4</i>	Fourth part of the name
in	<i>name_item5</i>	Fifth part of the name
in	<i>name_item6</i>	Sixth part of the name
in	<i>name_item7</i>	Seventh part of the name

Definition at line 167 of file `ref_frame_set_name.cc`.

References `name`, and `set_name()`.

8.5.3.28 set_owner()

```
void jeod::RefFrame::set_owner (
    RefFrameOwner * new_owner ) [inline], [virtual]
```

Set the owner of this frame.

Parameters

in	<i>new_owner</i>	New owner
----	------------------	-----------

Definition at line 99 of file `ref_frame_inline.hh`.

References `owner`.

8.5.3.29 set_timestamp()

```
void jeod::RefFrame::set_timestamp (
    double time ) [inline], [virtual]
```

Set the update time of this frame.

Parameters

in	<i>time</i>	Time Units: s
----	-------------	------------------

Definition at line 151 of file `ref_frame_inline.hh`.

References `update_time`.

8.5.3.30 timestamp()

```
double jeod::RefFrame::timestamp (
    void ) const [inline], [virtual]
```

Return the update time of this frame.

Returns

Time of last update
Units: s

Definition at line 164 of file `ref_frame_inline.hh`.

References `update_time`.

8.5.3.31 transplant_node()

```
void jeod::RefFrame::transplant_node (
    RefFrame & new_parent ) [virtual]
```

Move a node to a different place in the tree, keeping the state with respect to the root frame constant.

Parameters

in	<i>new_parent</i>	New parent frame
----	-------------------	------------------

Definition at line 102 of file `ref_frame.cc`.

References `compute_relative_state()`, `links`, `jeod::TreeLinks< Links, Container, Messages >::reattach()`, and `state`.

8.5.4 Friends And Related Function Documentation

8.5.4.1 `init_attrjeod__RefFrame`

```
void init_attrjeod__RefFrame ( ) [friend]
```

8.5.4.2 `InputProcessor`

```
friend class InputProcessor [friend]
```

Definition at line 101 of file `ref_frame.hh`.

8.5.4.3 `RefFrameLinks`

```
friend class RefFrameLinks [friend]
```

Definition at line 103 of file `ref_frame.hh`.

8.5.5 Field Documentation

8.5.5.1 `links`

```
RefFrameLinks jeod::RefFrame::links [protected]
```

Specifies the parent/child/sibling linkages between frames.

`trick_units(-)`

Definition at line 128 of file `ref_frame.hh`.

Referenced by `add_child()`, `compute_position_from()`, `compute_pred_rel_state()`, `compute_relative_state()`, `compute_state_wrt_pred()`, `find_last_common_index()`, `find_last_common_node()`, `get_parent()`, `get_root()`, `is_↔progeny_of()`, `make_root()`, `remove_from_parent()`, `reset_parent()`, `transplant_node()`, and `~RefFrame()`.

8.5.5.2 name

`std::string jeod::RefFrame::name` [protected]

The identifier for this reference frame.

`trick_units(-)`

Definition at line 118 of file `ref_frame.hh`.

Referenced by `compute_position_from()`, `compute_pred_rel_state()`, `compute_relative_state()`, `compute_state_wrt_pred()`, `get_name()`, and `set_name()`.

8.5.5.3 owner

`RefFrameOwner* jeod::RefFrame::owner` [protected]

The object that "owns" this frame.

`trick_units(-)`

Definition at line 123 of file `ref_frame.hh`.

Referenced by `get_owner()`, `set_active_status()`, and `set_owner()`.

8.5.5.4 state

`RefFrameState jeod::RefFrame::state`

The translational and rotational state of the reference frame with respect to its parent.

`trick_units(-)`

Definition at line 111 of file `ref_frame.hh`.

Referenced by `compute_position_from()`, `compute_pred_rel_state()`, `compute_state_wrt_pred()`, and `transplant_node()`.

8.5.5.5 update_time

`double jeod::RefFrame::update_time` [protected]

The time that the frame was last updated, dynamic time seconds.

`trick_units(s)`

Definition at line 133 of file `ref_frame.hh`.

Referenced by `set_timestamp()`, and `timestamp()`.

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

- [ref_frame.hh](#)
- [ref_frame_inline.hh](#)
- [ref_frame.cc](#)
- [ref_frame_compute_relative_state.cc](#)
- [ref_frame_set_name.cc](#)

8.6 jeod::RefFrameItems Class Reference

Identify which aspects of a reference frame's state have been set.

```
#include <ref_frame_items.hh>
```

Public Types

- enum [Items](#) {
[No_Items](#) = 0, [Pos](#) = 1, [Vel](#) = 2, [Pos_Vel](#) = 3,
[Att](#) = 4, [Pos_Att](#) = 5, [Vel_Att](#) = 6, [Pos_Vel_Att](#) = 7,
[Rate](#) = 8, [Pos_Rate](#) = 9, [Vel_Rate](#) = 10, [Pos_Vel_Rate](#) = 11,
[Att_Rate](#) = 12, [Pos_Att_Rate](#) = 13, [Vel_Att_Rate](#) = 14, [Pos_Vel_Att_Rate](#) = 15 }

The [Items](#) enumeration identifies the major items that can be set in a [RefFrameState](#) structure – position, velocity, attitude, and attitude rate.

Public Member Functions

- [RefFrameItems](#) (void)
Construct a [RefFrameItems](#) object.
- [RefFrameItems](#) ([Items](#) new_value)
Construct a [RefFrameItems](#) object.
- [Items](#) get (void) const
Get the value of a [RefFrameItems](#).
- bool [contains](#) ([Items](#) test_items) const
Determine if specified aspects of a [RefFrameItems](#) are set.
- bool [equals](#) ([Items](#) test_items) const
Determine whether a [RefFrameItems](#) equals the specified aspects.
- bool [is_empty](#) (void) const
Determine whether a [RefFrameItems](#) has nothing set.
- bool [is_full](#) (void) const
Determine whether a [RefFrameItems](#) has all bits set.
- [Items](#) set ([Items](#) new_value)
Set the value of a [RefFrameItems](#).
- [Items](#) add ([Items](#) new_items)
Set aspects of a [RefFrameItems](#).
- [Items](#) remove ([Items](#) old_items)
Clear aspects of a [RefFrameItems](#).
- const char * [to_string](#) (void) const
Return a string indicating the set items.

Static Public Member Functions

- static const char * [to_string](#) ([Items](#) test_items)
Return a string indicating the set items.

Data Fields

- [Items](#) value
Indicates which aspects of a [RefFrameState](#) have been set.

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameItems](#) ()

8.6.1 Detailed Description

Identify which aspects of a reference frame's state have been set.

The aspects that are managed are the position, velocity, attitude, and attitude rate.

Definition at line 83 of file `ref_frame_items.hh`.

8.6.2 Member Enumeration Documentation

8.6.2.1 Items

```
enum jeod::RefFrameItems::Items
```

The Items enumeration identifies the major items that can be set in a [RefFrameState](#) structure – position, velocity, attitude, and attitude rate.

The enumeration values are implemented as bit flags. The four basic items, position, velocity, attitude, and rate, have values of 1, 2, 4, and 8, respectively. Combinations thereof have values corresponding to the bitwise or of the basic components.

Enumerator

No_Items	Nothing set.
Pos	Position.
Vel	Velocity.
Pos_Vel	Position + velocity.
Att	Attitude.
Pos_Att	Position + attitude.
Vel_Att	Velocity + attitude.
Pos_Vel_Att	Position + velocity + attitude.
Rate	Attitude rate.
Pos_Rate	Position + rate.
Vel_Rate	Velocity + rate.
Pos_Vel_Rate	Position + velocity + rate.
Att_Rate	Attitude + Rate.
Pos_Att_Rate	Position + attitude + Rate.
Vel_Att_Rate	Velocity + attitude + Rate.
Pos_Vel_Att_Rate	Position + velocity + attitude + Rate.

Definition at line 100 of file `ref_frame_items.hh`.

8.6.3 Constructor & Destructor Documentation

8.6.3.1 RefFrameItems() [1/2]

```
jeod::RefFrameItems::RefFrameItems (
    void )
```

Construct a [RefFrameItems](#) object.

Definition at line 70 of file `ref_frame_items.cc`.

References `No_Items`, and `value`.

8.6.3.2 RefFrameItems() [2/2]

```
jeod::RefFrameItems::RefFrameItems (
    Items new_value )
```

Construct a [RefFrameItems](#) object.

Parameters

in	<i>new_value</i>	Initial value
----	------------------	---------------

Definition at line 81 of file `ref_frame_items.cc`.

References `value`.

8.6.4 Member Function Documentation

8.6.4.1 add()

```
RefFrameItems::Items jeod::RefFrameItems::add (
    RefFrameItems::Items new_items ) [inline]
```

Set aspects of a [RefFrameItems](#).

Returns

Updated value

Parameters

in	<i>new_items</i>	Items to add
----	------------------	--------------

Definition at line 163 of file `ref_frame_items_inline.hh`.

References value.

8.6.4.2 contains()

```
bool jeod::RefFrameItems::contains (
    RefFrameItems::Items test_items ) const [inline]
```

Determine if specified aspects of a [RefFrameItems](#) are set.

Returns

Are specified items set?

Parameters

in	<i>test_items</i>	Test items
----	-------------------	------------

Definition at line 93 of file `ref_frame_items_inline.hh`.

References value.

8.6.4.3 equals()

```
bool jeod::RefFrameItems::equals (
    RefFrameItems::Items test_items ) const [inline]
```

Determine whether a [RefFrameItems](#) equals the specified aspects.

Returns

Exact equality?

Parameters

in	<i>test_items</i>	Test items
----	-------------------	------------

Definition at line 109 of file `ref_frame_items_inline.hh`.

References value.

8.6.4.4 get()

```
RefFrameItems::Items jeod::RefFrameItems::get (  
    void ) const [inline]
```

Get the value of a [RefFrameItems](#).

Returns

Current value

Definition at line 79 of file `ref_frame_items_inline.hh`.

References `value`.

8.6.4.5 is_empty()

```
bool jeod::RefFrameItems::is_empty (  
    void ) const [inline]
```

Determine whether a [RefFrameItems](#) has nothing set.

Returns

Nothing set?

Definition at line 122 of file `ref_frame_items_inline.hh`.

References `No_Items`, and `value`.

8.6.4.6 is_full()

```
bool jeod::RefFrameItems::is_full (  
    void ) const [inline]
```

Determine whether a [RefFrameItems](#) has all bits set.

Returns

Fully set?

Definition at line 135 of file `ref_frame_items_inline.hh`.

References `Pos_Vel_Att_Rate`, and `value`.

8.6.4.7 remove()

```
RefFrameItems::Items jeod::RefFrameItems::remove (  
    RefFrameItems::Items old_items ) [inline]
```

Clear aspects of a [RefFrameItems](#).

Returns

Updated value

Parameters

in	<i>old_items</i>	Items to remove
----	------------------	-----------------

Definition at line 179 of file `ref_frame_items_inline.hh`.

References `Pos_Vel_Att_Rate`, and `value`.

8.6.4.8 set()

```
RefFrameItems::Items jeod::RefFrameItems::set (
    RefFrameItems::Items new_value ) [inline]
```

Set the value of a [RefFrameItems](#).

Returns

Updated value

Parameters

in	<i>new_value</i>	New value
----	------------------	-----------

Definition at line 149 of file `ref_frame_items_inline.hh`.

References `value`.

8.6.4.9 to_string() [1/2]

```
const char * jeod::RefFrameItems::to_string (
    Items test_items ) [static]
```

Return a string indicating the set items.

Returns

Set items, by name

Parameters

in	<i>test_items</i>	Items enum value
----	-------------------	------------------

Definition at line 39 of file `ref_frame_items.cc`.

References Att, Att_Rate, No_Items, Pos, Pos_Att, Pos_Att_Rate, Pos_Rate, Pos_Vel, Pos_Vel_Att, Pos_Vel_Att_Rate, Pos_Vel_Rate, Rate, Vel, Vel_Att, Vel_Att_Rate, and Vel_Rate.

8.6.4.10 to_string() [2/2]

```
const char * jeod::RefFrameItems::to_string (
    void ) const
```

Return a string indicating the set items.

Returns

Set items, by name

Definition at line 93 of file ref_frame_items.cc.

References value.

8.6.5 Friends And Related Function Documentation

8.6.5.1 init_attrjeod__RefFrameItems

```
void init_attrjeod__RefFrameItems ( ) [friend]
```

8.6.5.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 85 of file ref_frame_items.hh.

8.6.6 Field Documentation

8.6.6.1 value

Items jeod::RefFrameItems::value

Indicates which aspects of a [RefFrameState](#) have been set.

trick_units(—)

Definition at line 133 of file `ref_frame_items.hh`.

Referenced by `add()`, `contains()`, `equals()`, `get()`, `is_empty()`, `is_full()`, `RefFrameItems()`, `remove()`, `set()`, and `to_string()`.

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

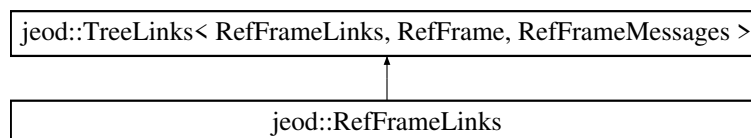
- [ref_frame_items.hh](#)
- [ref_frame_items_inline.hh](#)
- [ref_frame_items.cc](#)

8.7 jeod::RefFrameLinks Class Reference

Encapsulates the links between reference frames.

```
#include <ref_frame_links.hh>
```

Inheritance diagram for `jeod::RefFrameLinks`:



Public Member Functions

- [RefFrameLinks](#) ([RefFrame](#) &container_in)
Non-default constructor.
- virtual [~RefFrameLinks](#) (void)
Destructor.

Private Member Functions

- [RefFrameLinks](#) (void)
Not implemented.
- [RefFrameLinks](#) (const [RefFrameLinks](#) &)
Not implemented.
- void [operator=](#) (const [RefFrameLinks](#) &)
Not implemented.

Static Private Attributes

- static const unsigned int `default_path_size` = 4

Friends

- class `InputProcessor`
- void `init_attrjeod__RefFrameLinks` ()

Additional Inherited Members

8.7.1 Detailed Description

Encapsulates the links between reference frames.

Assumptions and Limitations

- Classes that use this class must keep the tree structure intact.

Definition at line 92 of file `ref_frame_links.hh`.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 RefFrameLinks() [1/3]

```
jeod::RefFrameLinks::RefFrameLinks (
    RefFrame & container_in ) [inline]
```

Non-default constructor.

Parameters

<code>container↔ _in</code>	The <code>RefFrame</code> object that contains this object.
---------------------------------	---

Definition at line 106 of file `ref_frame_links.hh`.

8.7.2.2 ~RefFrameLinks()

```
virtual jeod::RefFrameLinks::~~RefFrameLinks (
    void ) [inline], [virtual]
```


Destructor.

Definition at line 115 of file ref_frame_links.hh.

8.7.2.3 RefFrameLinks() [2/3]

```
jeod::RefFrameLinks::RefFrameLinks (  
    void ) [private]
```

Not implemented.

8.7.2.4 RefFrameLinks() [3/3]

```
jeod::RefFrameLinks::RefFrameLinks (  
    const RefFrameLinks & ) [private]
```

Not implemented.

8.7.3 Member Function Documentation

8.7.3.1 operator=()

```
void jeod::RefFrameLinks::operator= (  
    const RefFrameLinks & ) [private]
```

Not implemented.

8.7.4 Friends And Related Function Documentation

8.7.4.1 init_attrjeod__RefFrameLinks

```
void init_attrjeod__RefFrameLinks ( ) [friend]
```

8.7.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 96 of file `ref_frame_links.hh`.

8.7.5 Field Documentation

8.7.5.1 default_path_size

```
const unsigned int jeod::RefFrameLinks::default_path_size = 4 [static], [private]
```

Definition at line 121 of file `ref_frame_links.hh`.

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

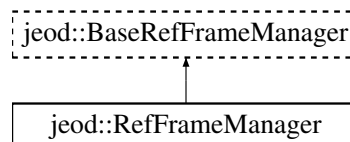
- [ref_frame_links.hh](#)

8.8 jeod::RefFrameManager Class Reference

The [RefFrameManager](#) class manages the reference frames in a simulation.

```
#include <ref_frame_manager.hh>
```

Inheritance diagram for `jeod::RefFrameManager`:



Public Member Functions

- [RefFrameManager](#) ()
RefFrameManager default constructor.
- virtual [~RefFrameManager](#) ()
RefFrameManager destructor.
- virtual void [add_ref_frame](#) ([RefFrame](#) &ref_frame)
Add a reference frame to the reference frame registry.
- virtual void [remove_ref_frame](#) ([RefFrame](#) &ref_frame)
Remove a reference frame from the reference frame registry.
- virtual [RefFrame](#) * [find_ref_frame](#) (const char *name) const
Find the reference frame with the given name.
- virtual [RefFrame](#) * [find_ref_frame](#) (const char *prefix, const char *suffix) const

- Find the reference frame with the dot-conjoined name "\${prefix}.\${suffix}".*

 - virtual void [check_ref_frame_ownership](#) (void) const

Check that each active reference frame has an owner.
- virtual void [reset_tree_root_node](#) ()

Reset the root node in anticipation of rebuilding the entire tree.
- virtual void [add_frame_to_tree](#) (RefFrame &ref_frame, RefFrame *parent)

Insert a reference frame in the reference frame tree.
- virtual void [subscribe_to_frame](#) (const char *frame_name)

Subscribe to a reference frame, with the frame specified by name.
- virtual void [subscribe_to_frame](#) (RefFrame &frame)

Subscribe to a reference frame, with the frame specified as an argument.
- virtual void [unsubscribe_to_frame](#) (const char *frame_name)

Remove subscription to a reference frame, with the frame specified by name.
- virtual void [unsubscribe_to_frame](#) (RefFrame &frame)

Remove subscription to a reference frame, with the frame specified as an argument.
- virtual bool [frame_is_subscribed](#) (const char *frame_name)

Checks whether frame has subscribers; frame specified by name.
- virtual bool [frame_is_subscribed](#) (RefFrame &frame)

Checks whether frame has subscribers; frame provided as an argument.

Protected Member Functions

- bool [validate_name](#) (const char *file, unsigned int line, const char *variable_value, const char *variable_type, const char *variable_name) const
- Check whether a name is trivially valid/invalid.*

Protected Attributes

- RefFrame * [root_node](#)
- The root node of the reference frame tree.*
- JeodPointerVector< [RefFrame](#) >::type [ref_frames](#)
- List of reference frames.*

Private Member Functions

- [RefFrameManager](#) (const [RefFrameManager](#) &)
- Not implemented.*
- [RefFrameManager](#) & [operator=](#) (const [RefFrameManager](#) &)
- Not implemented.*

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameManager](#) ()

8.8.1 Detailed Description

The [RefFrameManager](#) class manages the reference frames in a simulation.

This class is the base class for the EphemeridesManager and DynManager classes. Those derived classes add functionality to this class.

Definition at line 88 of file `ref_frame_manager.hh`.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 RefFrameManager() [1/2]

```
jeod::RefFrameManager::RefFrameManager (  
    void )
```

[RefFrameManager](#) default constructor.

Definition at line 45 of file `ref_frame_manager.cc`.

References `ref_frames`.

8.8.2.2 ~RefFrameManager()

```
jeod::RefFrameManager::~~RefFrameManager (  
    void ) [virtual]
```

[RefFrameManager](#) destructor.

Definition at line 61 of file `ref_frame_manager.cc`.

References `ref_frames`.

8.8.2.3 RefFrameManager() [2/2]

```
jeod::RefFrameManager::RefFrameManager (  
    const RefFrameManager & ) [private]
```

Not implemented.

8.8.3 Member Function Documentation

8.8.3.1 add_frame_to_tree()

```
void jeod::RefFrameManager::add_frame_to_tree (  
    RefFrame & ref_frame,  
    RefFrame * parent ) [virtual]
```

Insert a reference frame in the reference frame tree.

Parameters

<i>ref_frame</i>	Reference frame to be added to the ref frame tree.
<i>parent</i>	Parent frame

Implements [jeod::BaseRefFrameManager](#).

Definition at line 242 of file `ref_frame_manager.cc`.

References `jeod::RefFrame::add_child()`, `jeod::RefFrame::make_root()`, and `root_node`.

8.8.3.2 add_ref_frame()

```
void jeod::RefFrameManager::add_ref_frame (
    RefFrame & ref_frame ) [virtual]
```

Add a reference frame to the reference frame registry.

Parameters

<i>ref_frame</i>	Reference frame to be added.
------------------	------------------------------

Implements [jeod::BaseRefFrameManager](#).

Definition at line 78 of file `ref_frame_manager.cc`.

References `jeod::RefFrameMessages::duplicate_entry`, `find_ref_frame()`, `jeod::RefFrame::get_name()`, `ref_frames`, and `validate_name()`.

8.8.3.3 check_ref_frame_ownership()

```
void jeod::RefFrameManager::check_ref_frame_ownership (
    void ) const [virtual]
```

Check that each active reference frame has an owner.

Implements [jeod::BaseRefFrameManager](#).

Definition at line 206 of file `ref_frame_manager.cc`.

References `jeod::RefFrame::get_name()`, `jeod::RefFrame::get_owner()`, `jeod::RefFrameMessages::inconsistent_↔` setup, `jeod::Subscription::is_active()`, and `ref_frames`.

8.8.3.4 find_ref_frame() [1/2]

```
RefFrame * jeod::RefFrameManager::find_ref_frame (
    const char * name ) const [virtual]
```

Find the reference frame with the given name.

Parameters

<i>name</i>	Reference frame name
-------------	----------------------

Returns

Found reference frame, or NULL if not found

Implements [jeod::BaseRefFrameManager](#).

Definition at line 146 of file `ref_frame_manager.cc`.

References `jeod::RefFrame::get_name()`, and `ref_frames`.

Referenced by `add_ref_frame()`, `frame_is_subscribed()`, `subscribe_to_frame()`, and `unsubscribe_to_frame()`.

8.8.3.5 find_ref_frame() [2/2]

```
RefFrame * jeod::RefFrameManager::find_ref_frame (
    const char * prefix,
    const char * suffix ) const [virtual]
```

Find the reference frame with the dot-conjoined name "\${prefix}.\${suffix}".

Parameters

<i>prefix</i>	Reference frame name prefix
<i>suffix</i>	Reference frame name suffix

Returns

Found reference frame, or NULL if not found

Implements [jeod::BaseRefFrameManager](#).

Definition at line 175 of file `ref_frame_manager.cc`.

References `jeod::RefFrame::get_name()`, and `ref_frames`.

8.8.3.6 frame_is_subscribed() [1/2]

```
bool jeod::RefFrameManager::frame_is_subscribed (
    const char * frame_name ) [virtual]
```

Checks whether frame has subscribers; frame specified by name.

Parameters

<i>frame_name</i>	Name of reference frame
-------------------	-------------------------

Returns

True if the frame has subscribers; false otherwise.

Implements [jeod::BaseRefFrameManager](#).

Definition at line 381 of file `ref_frame_manager.cc`.

References `find_ref_frame()`, `jeod::RefFrameMessages::invalid_name`, and `validate_name()`.

8.8.3.7 `frame_is_subscribed()` [2/2]

```
bool jeod::RefFrameManager::frame_is_subscribed (
    RefFrame & frame ) [virtual]
```

Checks whether frame has subscribers; frame provided as an argument.

Parameters

<i>frame</i>	The reference frame
--------------	---------------------

Returns

True if the frame has subscribers; false otherwise.

Implements [jeod::BaseRefFrameManager](#).

Definition at line 412 of file `ref_frame_manager.cc`.

References `jeod::Subscription::subscriptions()`.

8.8.3.8 `operator=()`

```
RefFrameManager& jeod::RefFrameManager::operator= (
    const RefFrameManager & ) [private]
```

Not implemented.

8.8.3.9 `remove_ref_frame()`

```
void jeod::RefFrameManager::remove_ref_frame (
    RefFrame & ref_frame ) [virtual]
```

Remove a reference frame from the reference frame registry.

Parameters

<i>ref_frame</i>	Reference frame to be removed.
------------------	--------------------------------

Implements [jeod::BaseRefFrameManager](#).

Definition at line 122 of file `ref_frame_manager.cc`.

References `jeod::RefFrame::get_name()`, `ref_frames`, and `jeod::RefFrameMessages::removal_failed`.

8.8.3.10 reset_tree_root_node()

```
void jeod::RefFrameManager::reset_tree_root_node (
    void ) [virtual]
```

Reset the root node in anticipation of rebuilding the entire tree.

Implements [jeod::BaseRefFrameManager](#).

Definition at line 229 of file `ref_frame_manager.cc`.

References `root_node`.

8.8.3.11 subscribe_to_frame() [1/2]

```
void jeod::RefFrameManager::subscribe_to_frame (
    const char * frame_name ) [virtual]
```

Subscribe to a reference frame, with the frame specified by name.

Assumptions and limitations:

- A subscriber should not double-subscribe to a frame.

Parameters

<i>frame_name</i>	Name of reference frame
-------------------	-------------------------

Implements [jeod::BaseRefFrameManager](#).

Definition at line 275 of file `ref_frame_manager.cc`.

References `find_ref_frame()`, `jeod::RefFrameMessages::invalid_name`, and `validate_name()`.

8.8.3.12 subscribe_to_frame() [2/2]

```
void jeod::RefFrameManager::subscribe_to_frame (
    RefFrame & frame ) [virtual]
```

Subscribe to a reference frame, with the frame specified as an argument.

Assumptions and limitations:

- A subscriber should not double-subscribe to a frame.

Parameters

<i>frame</i>	The reference frame to be subscribed to.
--------------	--

Implements [jeod::BaseRefFrameManager](#).

Definition at line 310 of file `ref_frame_manager.cc`.

References `jeod::Subscription::subscribe()`.

8.8.3.13 unsubscribe_to_frame() [1/2]

```
void jeod::RefFrameManager::unsubscribe_to_frame (
    const char * frame_name ) [virtual]
```

Remove subscription to a reference frame, with the frame specified by name.

Assumptions and limitations:

- The caller is subscribed to the frame.

Parameters

<i>frame_name</i>	Name of reference frame
-------------------	-------------------------

Implements [jeod::BaseRefFrameManager](#).

Definition at line 326 of file `ref_frame_manager.cc`.

References `find_ref_frame()`, `jeod::RefFrameMessages::invalid_name`, and `validate_name()`.

8.8.3.14 unsubscribe_to_frame() [2/2]

```
void jeod::RefFrameManager::unsubscribe_to_frame (
    RefFrame & frame ) [virtual]
```

Remove subscription to a reference frame, with the frame specified as an argument.

Assumptions and limitations:

- The caller is subscribed to the frame.

Parameters

<i>frame</i>	The reference frame
--------------	---------------------

Implements [jeod::BaseRefFrameManager](#).

Definition at line 361 of file `ref_frame_manager.cc`.

References [jeod::RefFrame::get_name\(\)](#), [jeod::RefFrameMessages::invalid_item](#), [jeod::Subscription::subscriptions\(\)](#), and [jeod::Subscription::unsubscribe\(\)](#).

8.8.3.15 validate_name()

```
bool jeod::RefFrameManager::validate_name (
    const char * file,
    unsigned int line,
    const char * variable_value,
    const char * variable_type,
    const char * variable_name ) const [protected]
```

Check whether a name is trivially valid/invalid.

Parameters

<i>file</i>	Usually FILE
<i>line</i>	Usually LINE
<i>variable_value</i>	Value to check
<i>variable_type</i>	Variable description
<i>variable_name</i>	Variable name

Returns

True if the name is valid, false if invalid.

Definition at line 434 of file `ref_frame_manager.cc`.

References [jeod::RefFrameMessages::invalid_name](#), and [jeod::RefFrameMessages::null_pointer](#).

Referenced by [add_ref_frame\(\)](#), [frame_is_subscribed\(\)](#), [subscribe_to_frame\(\)](#), and [unsubscribe_to_frame\(\)](#).

8.8.4 Friends And Related Function Documentation

8.8.4.1 init_attrjeod__RefFrameManager

```
void init_attrjeod__RefFrameManager ( ) [friend]
```

8.8.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 90 of file `ref_frame_manager.hh`.

8.8.5 Field Documentation

8.8.5.1 ref_frames

```
JeodPointerVector<RefFrame>::type jeod::RefFrameManager::ref_frames [protected]
```

List of reference frames.

```
trick_io(**)
```

Definition at line 167 of file `ref_frame_manager.hh`.

Referenced by `add_ref_frame()`, `check_ref_frame_ownership()`, `find_ref_frame()`, `RefFrameManager()`, `remove_ref_frame()`, and `~RefFrameManager()`.

8.8.5.2 root_node

```
RefFrame* jeod::RefFrameManager::root_node [protected]
```

The root node of the reference frame tree.

This reference frame is the true inertial frame of the simulation.`trick_units(-)`

Definition at line 162 of file `ref_frame_manager.hh`.

Referenced by `add_frame_to_tree()`, and `reset_tree_root_node()`.

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

- [ref_frame_manager.hh](#)
- [ref_frame_manager.cc](#)

8.9 jeod::RefFrameMessages Class Reference

Declares messages associated with the reference frames model.

```
#include <ref_frame_messages.hh>
```

Static Public Attributes

- static char const * [attach_info](#) = "utils/ref_frames/" "attach_info"
Issued to provide information regarding an attachment.
- static char const * [duplicate_entry](#) = "utils/ref_frames/" "duplicate_entry"
Issued when a duplicate reference frame is detected (name or address).
- static char const * [inconsistent_setup](#) = "utils/ref_frames/" "inconsistent_setup"
Issued when some inconsistency is detected.
- static char const * [internal_error](#) = "utils/ref_frames/" "internal_error"
Error issued when some internal error occurred.
- static char const * [invalid_attach](#) = "utils/ref_frames/" "invalid_attach"
Issued when an attachment cannot be performed as requested.
- static char const * [invalid_detach](#) = "utils/ref_frames/" "invalid_detach"
Issued when a detachment cannot be performed as requested.
- static char const * [invalid_enum](#) = "utils/ref_frames/" "invalid_enum"
Issued when a enum value is not one of the enumerated values.
- static char const * [invalid_item](#) = "utils/ref_frames/" "invalid_item"
Issued when something other than an enum, name, or node is invalid.
- static char const * [invalid_name](#) = "utils/ref_frames/" "invalid_name"
Issued when a name is invalid – NULL, empty, a duplicate, ...
- static char const * [invalid_node](#) = "utils/ref_frames/" "invalid_node"
Issued when a node does not have expected linkages.
- static char const * [null_pointer](#) = "utils/ref_frames/" "null_pointer"
Issued when a pointer that is null should be non-null.
- static char const * [subscription_error](#) = "utils/ref_frames/" "subscription_error"
Error issued when a problem is detected in the subscription model.
- static char const * [removal_failed](#) = "utils/ref_frames/" "removal_failed"
Error issued when a removal cannot be performed because the frame is not registered.

Private Member Functions

- [RefFrameMessages](#) (void)
- [RefFrameMessages](#) (const [RefFrameMessages](#) &)
- [RefFrameMessages](#) & operator= (const [RefFrameMessages](#) &)

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameMessages](#) ()

8.9.1 Detailed Description

Declares messages associated with the reference frames model.

Definition at line 83 of file ref_frame_messages.hh.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 RefFrameMessages() [1/2]

```
jeod::RefFrameMessages::RefFrameMessages (  
    void ) [private]
```

8.9.2.2 RefFrameMessages() [2/2]

```
jeod::RefFrameMessages::RefFrameMessages (  
    const RefFrameMessages & ) [private]
```

8.9.3 Member Function Documentation

8.9.3.1 operator=()

```
RefFrameMessages& jeod::RefFrameMessages::operator= (  
    const RefFrameMessages & ) [private]
```

8.9.4 Friends And Related Function Documentation

8.9.4.1 init_attrjeod__RefFrameMessages

```
void init_attrjeod__RefFrameMessages ( ) [friend]
```

8.9.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 86 of file ref_frame_messages.hh.

8.9.5 Field Documentation

8.9.5.1 attach_info

```
char const * jeod::RefFrameMessages::attach_info = "utils/ref_frames/" "attach_info" [static]
```

Issued to provide information regarding an attachment.

trick_units(—)

Definition at line 94 of file ref_frame_messages.hh.

8.9.5.2 duplicate_entry

```
char const * jeod::RefFrameMessages::duplicate_entry = "utils/ref_frames/" "duplicate_entry" [static]
```

Issued when a duplicate reference frame is detected (name or address).

trick_units(—)

Definition at line 99 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::add_ref_frame().

8.9.5.3 inconsistent_setup

```
char const * jeod::RefFrameMessages::inconsistent_setup = "utils/ref_frames/" "inconsistent_↔  
setup" [static]
```

Issued when some inconsistency is detected.

trick_units(—)

Definition at line 104 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::check_ref_frame_ownership().

8.9.5.4 internal_error

```
char const * jeod::RefFrameMessages::internal_error = "utils/ref_frames/" "internal_error"  
[static]
```

Error issued when some internal error occurred.

These errors should never happen.`trick_units(-)`

Definition at line 110 of file `ref_frame_messages.hh`.

8.9.5.5 invalid_attach

```
char const * jeod::RefFrameMessages::invalid_attach = "utils/ref_frames/" "invalid_attach"  
[static]
```

Issued when an attachment cannot be performed as requested.

`trick_units(-)`

Definition at line 115 of file `ref_frame_messages.hh`.

8.9.5.6 invalid_detach

```
char const * jeod::RefFrameMessages::invalid_detach = "utils/ref_frames/" "invalid_detach"  
[static]
```

Issued when a detachment cannot be performed as requested.

`trick_units(-)`

Definition at line 120 of file `ref_frame_messages.hh`.

8.9.5.7 invalid_enum

```
char const * jeod::RefFrameMessages::invalid_enum = "utils/ref_frames/" "invalid_enum" [static]
```

Issued when a enum value is not one of the enumerated values.

`trick_units(-)`

Definition at line 125 of file `ref_frame_messages.hh`.

8.9.5.8 invalid_item

```
char const * jeod::RefFrameMessages::invalid_item = "utils/ref_frames/" "invalid_item" [static]
```

Issued when something other than an enum, name, or node is invalid.

trick_units(-)

Definition at line 130 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::unsubscribe_to_frame().

8.9.5.9 invalid_name

```
char const * jeod::RefFrameMessages::invalid_name = "utils/ref_frames/" "invalid_name" [static]
```

Issued when a name is invalid – NULL, empty, a duplicate, ...

trick_units(-)

Definition at line 135 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::frame_is_subscribed(), jeod::RefFrameManager::subscribe_to_frame(), jeod::RefFrameManager::unsubscribe_to_frame(), and jeod::RefFrameManager::validate_name().

8.9.5.10 invalid_node

```
char const * jeod::RefFrameMessages::invalid_node = "utils/ref_frames/" "invalid_node" [static]
```

Issued when a node does not have expected linkages.

trick_units(-)

Definition at line 140 of file ref_frame_messages.hh.

Referenced by jeod::RefFrame::compute_position_from(), jeod::RefFrame::compute_pred_rel_state(), jeod::RefFrame::compute_relative_state(), and jeod::RefFrame::compute_state_wrt_pred().

8.9.5.11 null_pointer

```
char const * jeod::RefFrameMessages::null_pointer = "utils/ref_frames/" "null_pointer" [static]
```

Issued when a pointer that is null should be non-null.

trick_units(-)

Definition at line 145 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::validate_name().

8.9.5.12 removal_failed

```
char const * jeod::RefFrameMessages::removal_failed = "utils/ref_frames/" "removal_failed"
[static]
```

Error issued when a removal cannot be performed because the frame is not registered.

trick_units(-)

Definition at line 156 of file ref_frame_messages.hh.

Referenced by jeod::RefFrameManager::remove_ref_frame().

8.9.5.13 subscription_error

```
char const * jeod::RefFrameMessages::subscription_error = "utils/ref_frames/" "subscription_↵
error" [static]
```

Error issued when a problem is detected in the subscription model.

trick_units(-)

Definition at line 150 of file ref_frame_messages.hh.

Referenced by jeod::Subscription::activate(), jeod::Subscription::deactivate(), jeod::Subscription::subscribe(), and jeod::Subscription::unsubscribe().

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

- [ref_frame_messages.hh](#)
- [ref_frame_messages.cc](#)

8.10 jeod::RefFrameOwner Class Reference

Identify an object as an "owner" of a reference frame.

```
#include <ref_frame_interface.hh>
```

Public Member Functions

- [RefFrameOwner](#) ()
RefFrameOwner default constructor.
- virtual [~RefFrameOwner](#) ()
RefFrameOwner destructor.
- virtual void [note_frame_status_change](#) ([RefFrame](#) *frame)
Note that a reference frame has changed its active/inactive status.

8.10.1 Detailed Description

Identify an object as an "owner" of a reference frame.

This class is an interface – it has no member data. It instead defines minimal capabilities common to all things that can "own" a reference frame.

This interface class is one of the very few classes that JEOD uses in the form of multiple inheritance.

Definition at line 81 of file `ref_frame_interface.hh`.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 RefFrameOwner()

```
jeod::RefFrameOwner::RefFrameOwner ( ) [inline]
```

[RefFrameOwner](#) default constructor.

Definition at line 91 of file `ref_frame_interface.hh`.

8.10.2.2 ~RefFrameOwner()

```
virtual jeod::RefFrameOwner::~~RefFrameOwner ( ) [inline], [virtual]
```

[RefFrameOwner](#) destructor.

Definition at line 96 of file `ref_frame_interface.hh`.

8.10.3 Member Function Documentation

8.10.3.1 note_frame_status_change()

```
virtual void jeod::RefFrameOwner::note_frame_status_change (
    RefFrame * frame ) [inline], [virtual]
```

Note that a reference frame has changed its active/inactive status.

This default implementation does nothing.

Parameters

<i>frame</i>	Frame whose status has changed
--------------	--------------------------------

Definition at line 104 of file `ref_frame_interface.hh`.

Referenced by `jeod::RefFrame::set_active_status()`.

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

- [ref_frame_interface.hh](#)

8.11 jeod::RefFrameRot Class Reference

Represent the rotational aspects of a reference frame's state.

```
#include <ref_frame_state.hh>
```

Public Member Functions

- [RefFrameRot](#) (void)
Default constructor; initializes state to a null rotation.
- [RefFrameRot](#) (const [RefFrameRot](#) &source)
Copy constructor; initializes state to that of the source.
- [RefFrameRot](#) & [operator=](#) (const [RefFrameRot](#) &source)
Assignment operator; copies state from the source.
- [~RefFrameRot](#) (void)
Destructor; does nothing.
- void [initialize](#) ()
Initialize a [RefFrameRot](#) to a null offset.
- void [copy](#) (const [RefFrameRot](#) &source)
Initialize a [RefFrameRot](#) from a source state.
- void [compute_transformation](#) ()
Compute the transformation matrix from the left quaternion.
- void [compute_quaternion](#) ()
Compute the left quaternion from the transformation matrix.
- void [compute_ang_vel_unit](#) ()
Compute the angular velocity unit vector.
- void [compute_ang_vel_products](#) ()
Compute the angular velocity magnitude and unit vector.

Data Fields

- Quaternion [Q_parent_this](#)
Left transformation quaternion from the parent reference frame to the subject reference frame.
- double [T_parent_this](#) [3][3]
Transformation matrix from the parent reference frame to the subject reference frame.
- double [ang_vel_this](#) [3]
Angular velocity of the subject reference frame with respect to the parent reference frame expressed in subject reference frame coordinates.
- double [ang_vel_mag](#)
Magnitude of [ang_vel_this](#).
- double [ang_vel_unit](#) [3]
Unit vector in the direction of [ang_vel_this](#).

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameRot](#) ()

8.11.1 Detailed Description

Represent the rotational aspects of a reference frame's state.

Definition at line 127 of file `ref_frame_state.hh`.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 RefFrameRot() [1/2]

```
jeod::RefFrameRot::RefFrameRot (
    void ) [inline]
```

Default constructor; initializes state to a null rotation.

Definition at line 144 of file `ref_frame_state_inline.hh`.

References `initialize()`.

8.11.2.2 RefFrameRot() [2/2]

```
jeod::RefFrameRot::RefFrameRot (
    const RefFrameRot & source ) [inline]
```

Copy constructor; initializes state to that of the source.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 156 of file `ref_frame_state_inline.hh`.

References `copy()`.

8.11.2.3 ~RefFrameRot()

```
jeod::RefFrameRot::~~RefFrameRot (
    void ) [inline]
```

Destructor; does nothing.

Definition at line 167 of file ref_frame_state_inline.hh.

8.11.3 Member Function Documentation

8.11.3.1 compute_ang_vel_products()

```
void jeod::RefFrameRot::compute_ang_vel_products (
    void ) [inline]
```

Compute the angular velocity magnitude and unit vector.

Definition at line 250 of file ref_frame_state_inline.hh.

References ang_vel_mag, ang_vel_this, and compute_ang_vel_unit().

Referenced by jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_↔left(), and jeod::RefFrameState::incr_right().

8.11.3.2 compute_ang_vel_unit()

```
void jeod::RefFrameRot::compute_ang_vel_unit (
    void ) [inline]
```

Compute the angular velocity unit vector.

Assumptions and Limitations

- Angular velocity magnitude has already been computed.

Definition at line 234 of file ref_frame_state_inline.hh.

References ang_vel_mag, ang_vel_this, and ang_vel_unit.

Referenced by compute_ang_vel_products().

8.11.3.3 compute_quaternion()

```
void jeod::RefFrameRot::compute_quaternion (
    void ) [inline]
```

Compute the left quaternion from the transformation matrix.

Definition at line 220 of file ref_frame_state_inline.hh.

References Q_parent_this, and T_parent_this.

8.11.3.4 compute_transformation()

```
void jeod::RefFrameRot::compute_transformation (
    void ) [inline]
```

Compute the transformation matrix from the left quaternion.

Definition at line 209 of file ref_frame_state_inline.hh.

References Q_parent_this, and T_parent_this.

Referenced by jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_↵left(), and jeod::RefFrameState::incr_right().

8.11.3.5 copy()

```
void jeod::RefFrameRot::copy (
    const RefFrameRot & source ) [inline]
```

Initialize a [RefFrameRot](#) from a source state.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 194 of file ref_frame_state_inline.hh.

References ang_vel_mag, ang_vel_this, ang_vel_unit, Q_parent_this, and T_parent_this.

Referenced by jeod::RefFrameState::copy(), jeod::RefFrameState::incr_right(), operator=(), and RefFrameRot().

8.11.3.6 initialize()

```
void jeod::RefFrameRot::initialize (
    void ) [inline]
```

Initialize a [RefFrameRot](#) to a null offset.

Definition at line 178 of file ref_frame_state_inline.hh.

References ang_vel_mag, ang_vel_this, ang_vel_unit, Q_parent_this, and T_parent_this.

Referenced by jeod::RefFrameState::initialize(), jeod::RefFrameState::negate(), and RefFrameRot().

8.11.3.7 operator=()

```
RefFrameRot & jeod::RefFrameRot::operator= (
    const RefFrameRot & source )
```

Assignment operator; copies state from the source.

Returns

Pointer to this

Parameters

in	source	Source state
----	--------	--------------

Definition at line 142 of file ref_frame_state.cc.

References [copy\(\)](#).

8.11.4 Friends And Related Function Documentation

8.11.4.1 init_attrjeod__RefFrameRot

```
void init_attrjeod__RefFrameRot ( ) [friend]
```

8.11.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 129 of file ref_frame_state.hh.

8.11.5 Field Documentation

8.11.5.1 ang_vel_mag

```
double jeod::RefFrameRot::ang_vel_mag
```

Magnitude of ang_vel_this.

trick_units(rad/s)

Definition at line 155 of file ref_frame_state.hh.

Referenced by [compute_ang_vel_products\(\)](#), [compute_ang_vel_unit\(\)](#), [copy\(\)](#), [jeod::RefFrameState::decr_left\(\)](#), [jeod::RefFrameState::decr_right\(\)](#), [jeod::RefFrameState::incr_left\(\)](#), [jeod::RefFrameState::incr_right\(\)](#), [initialize\(\)](#), and [jeod::RefFrameState::negate\(\)](#).

8.11.5.2 ang_vel_this

```
double jeod::RefFrameRot::ang_vel_this[3]
```

Angular velocity of the subject reference frame with respect to the parent reference frame expressed in subject reference frame coordinates.

trick_units(rad/s)

Definition at line 150 of file ref_frame_state.hh.

Referenced by compute_ang_vel_products(), compute_ang_vel_unit(), jeod::RefFrame::compute_relative_state(), copy(), jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_left(), jeod::RefFrameState::incr_right(), initialize(), and jeod::RefFrameState::negate().

8.11.5.3 ang_vel_unit

```
double jeod::RefFrameRot::ang_vel_unit[3]
```

Unit vector in the direction of ang_vel_this.

trick_units(-)

Definition at line 160 of file ref_frame_state.hh.

Referenced by compute_ang_vel_unit(), jeod::RefFrame::compute_relative_state(), copy(), initialize(), and jeod::RefFrameState::negate().

8.11.5.4 Q_parent_this

```
Quaternion jeod::RefFrameRot::Q_parent_this
```

Left transformation quaternion from the parent reference frame to the subject reference frame.

trick_units(-)

Definition at line 138 of file ref_frame_state.hh.

Referenced by jeod::RefFrame::compute_position_from(), compute_quaternion(), jeod::RefFrame::compute_relative_state(), compute_transformation(), copy(), jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_left(), jeod::RefFrameState::incr_right(), initialize(), and jeod::RefFrameState::negate().

8.11.5.5 T_parent_this

```
double jeod::RefFrameRot::T_parent_this[3][3]
```

Transformation matrix from the parent reference frame to the subject reference frame.

trick_units(-)

Definition at line 144 of file ref_frame_state.hh.

Referenced by `jeod::RefFrame::compute_position_from()`, `compute_quaternion()`, `jeod::RefFrame::compute_relative_state()`, `compute_transformation()`, `copy()`, `jeod::RefFrameState::decr_left()`, `jeod::RefFrameState::decr_right()`, `jeod::RefFrameState::incr_left()`, `jeod::RefFrameState::incr_right()`, `initialize()`, and `jeod::RefFrameState::negate()`.

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

- [ref_frame_state.hh](#)
- [ref_frame_state_inline.hh](#)
- [ref_frame_state.cc](#)

8.12 jeod::RefFrameState Class Reference

Represent a reference frame's state.

```
#include <ref_frame_state.hh>
```

Public Member Functions

- [RefFrameState](#) ()
RefFrameState default constructor.
- [RefFrameState](#) (const [RefFrameState](#) &source)
RefFrameState copy constructor.
- [RefFrameState](#) & operator= (const [RefFrameState](#) &source)
Assignment operator; copies state from the source.
- [~RefFrameState](#) (void)
Destructor; does nothing.
- void [initialize](#) ()
Initialize a [RefFrameState](#) to a null offset.
- void [copy](#) (const [RefFrameState](#) &source)
Initialize a [RefFrameState](#) from a source state.
- void [negate](#) (const [RefFrameState](#) &source)
Copy a reference frame state, negated.
- void [incr_left](#) (const [RefFrameState](#) &s_ab)
Compute $S_A:C = S_A:B + S_B:C$, with this initially containing $S_B:C$, the supplied argument containing $S_A:B$, and the resultant composition of states stored in this.
- void [incr_right](#) (const [RefFrameState](#) &s_bc)
Compute $S_A:C = S_A:B + S_B:C$, with this initially containing $S_A:B$, the supplied argument containing $S_B:C$, and the resultant composition of states stored in this.
- void [decr_left](#) (const [RefFrameState](#) &s_ab)
Compute $S_B:C = (-S_A:B) + S_A:C$, with this initially containing $S_A:C$, the supplied argument containing $S_A:B$, and the resultant composition of states stored in this.
- void [decr_right](#) (const [RefFrameState](#) &s_bc)
Compute $S_A:B = S_A:C + (-S_B:C)$ with this initially containing $S_A:C$, the supplied argument containing $S_B:C$, and the resultant composition of states stored in this.

Data Fields

- [RefFrameTrans](#) `trans`
Translation state.
- [RefFrameRot](#) `rot`
Rotational state.

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameState](#) ()

8.12.1 Detailed Description

Represent a reference frame's state.

Definition at line 201 of file `ref_frame_state.hh`.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 [RefFrameState](#)() [1/2]

```
jeod::RefFrameState::RefFrameState (
    void )
```

[RefFrameState](#) default constructor.

Definition at line 156 of file `ref_frame_state.cc`.

References `initialize()`.

8.12.2.2 [RefFrameState](#)() [2/2]

```
jeod::RefFrameState::RefFrameState (
    const RefFrameState & source )
```

[RefFrameState](#) copy constructor.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 167 of file `ref_frame_state.cc`.

References `copy()`.

8.12.2.3 ~RefFrameState()

```
jeod::RefFrameState::~~RefFrameState (
    void ) [inline]
```

Destructor; does nothing.

Definition at line 263 of file `ref_frame_state_inline.hh`.

8.12.3 Member Function Documentation

8.12.3.1 copy()

```
void jeod::RefFrameState::copy (
    const RefFrameState & source ) [inline]
```

Initialize a [RefFrameState](#) from a source state.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 287 of file `ref_frame_state_inline.hh`.

References `jeod::RefFrameTrans::copy()`, `jeod::RefFrameRot::copy()`, `rot`, and `trans`.

Referenced by `jeod::RefFrame::compute_state_wrt_pred()`, `operator=()`, and `RefFrameState()`.

8.12.3.2 decr_left()

```
void jeod::RefFrameState::decr_left (
    const RefFrameState & s_ab )
```

Compute $S_B:C = (-S_A:B) + S_A:C$, with this initially containing $S_A:C$, the supplied argument containing $S_A:B$, and the resultant composition of states stored in this.

Parameters

in	<i>s_ab</i>	Left addend
----	-------------	-------------

Definition at line 430 of file ref_frame_state.cc.

References jeod::RefFrameRot::ang_vel_mag, jeod::RefFrameRot::ang_vel_this, jeod::RefFrameRot::compute_↵
_ang_vel_products(), jeod::RefFrameRot::compute_transformation(), jeod::RefFrameTrans::position, jeod::Ref↵
FrameRot::Q_parent_this, rot, jeod::RefFrameRot::T_parent_this, trans, and jeod::RefFrameTrans::velocity.

Referenced by jeod::RefFrame::compute_relative_state().

8.12.3.3 decr_right()

```
void jeod::RefFrameState::decr_right (
    const RefFrameState & s_bc )
```

Compute $S_A:B = S_A:C + (-S_B:C)$ with this initially containing $S_A:C$, the supplied argument containing $S_B:C$, and the resultant composition of states stored in this.

Parameters

in	s_bc	Left addend
----	------	-------------

Definition at line 500 of file ref_frame_state.cc.

References jeod::RefFrameRot::ang_vel_mag, jeod::RefFrameRot::ang_vel_this, jeod::RefFrameRot::compute_↵
_ang_vel_products(), jeod::RefFrameRot::compute_transformation(), jeod::RefFrameTrans::position, jeod::Ref↵
FrameRot::Q_parent_this, rot, jeod::RefFrameRot::T_parent_this, trans, and jeod::RefFrameTrans::velocity.

Referenced by jeod::RefFrame::compute_pred_rel_state().

8.12.3.4 incr_left()

```
void jeod::RefFrameState::incr_left (
    const RefFrameState & s_ab )
```

Compute $S_A:C = S_A:B + S_B:C$, with this initially containing $S_B:C$, the supplied argument containing $S_A:B$, and the resultant composition of states stored in this.

Parameters

in	s_ab	Left addend
----	------	-------------

Definition at line 257 of file ref_frame_state.cc.

References jeod::RefFrameRot::ang_vel_mag, jeod::RefFrameRot::ang_vel_this, jeod::RefFrameRot::compute_↵
_ang_vel_products(), jeod::RefFrameRot::compute_transformation(), jeod::RefFrameTrans::position, jeod::Ref↵
FrameRot::Q_parent_this, rot, jeod::RefFrameRot::T_parent_this, trans, and jeod::RefFrameTrans::velocity.

Referenced by jeod::RefFrame::compute_state_wrt_pred().

8.12.3.5 incr_right()

```
void jeod::RefFrameState::incr_right (
    const RefFrameState & s_bc )
```

Compute $S_A:C = S_A:B + S_B:C$, with this initially containing $S_A:B$, the supplied argument containing $S_B:C$, and the resultant composition of states stored in this.

Note that this function is untested, as it is not used in the Reference Frame Model at any point, and is only given here as a utility function.

Parameters

in	s_bc	Right addend
----	------	--------------

Definition at line 344 of file ref_frame_state.cc.

References jeod::RefFrameRot::ang_vel_mag, jeod::RefFrameRot::ang_vel_this, jeod::RefFrameRot::compute_↵
ang_vel_products(), jeod::RefFrameRot::compute_transformation(), jeod::RefFrameRot::copy(), jeod::RefFrame↵
Trans::position, jeod::RefFrameRot::Q_parent_this, rot, jeod::RefFrameRot::T_parent_this, trans, and jeod::Ref↵
FrameTrans::velocity.

8.12.3.6 initialize()

```
void jeod::RefFrameState::initialize (
    void ) [inline]
```

Initialize a [RefFrameState](#) to a null offset.

Definition at line 274 of file ref_frame_state_inline.hh.

References jeod::RefFrameTrans::initialize(), jeod::RefFrameRot::initialize(), rot, and trans.

Referenced by jeod::RefFrame::compute_relative_state(), and RefFrameState().

8.12.3.7 negate()

```
void jeod::RefFrameState::negate (
    const RefFrameState & source )
```

Copy a reference frame state, negated.

Parameters

in	source	Source state
----	--------	--------------

Definition at line 195 of file ref_frame_state.cc.

References jeod::RefFrameRot::ang_vel_mag, jeod::RefFrameRot::ang_vel_this, jeod::RefFrameRot::ang_vel_↵ unit, jeod::RefFrameRot::initialize(), jeod::RefFrameTrans::position, jeod::RefFrameRot::Q_parent_this, rot, jeod::↵ RefFrameRot::T_parent_this, trans, and jeod::RefFrameTrans::velocity.

Referenced by jeod::RefFrame::compute_pred_rel_state().

8.12.3.8 operator=()

```
RefFrameState & jeod::RefFrameState::operator= (
    const RefFrameState & source )
```

Assignment operator; copies state from the source.

Returns

Pointer to this

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 180 of file ref_frame_state.cc.

References copy().

8.12.4 Friends And Related Function Documentation

8.12.4.1 init_attrjeod__RefFrameState

```
void init_attrjeod__RefFrameState ( ) [friend]
```

8.12.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 203 of file ref_frame_state.hh.

8.12.5 Field Documentation

8.12.5.1 rot

[RefFrameRot](#) jeod::RefFrameState::rot

Rotational state.

trick_units(-)

Definition at line 215 of file ref_frame_state.hh.

Referenced by jeod::RefFrame::compute_position_from(), jeod::RefFrame::compute_relative_state(), copy(), decr_left(), decr_right(), incr_left(), incr_right(), initialize(), and negate().

8.12.5.2 trans

[RefFrameTrans](#) jeod::RefFrameState::trans

Translation state.

trick_units(-)

Definition at line 210 of file ref_frame_state.hh.

Referenced by jeod::RefFrame::compute_position_from(), jeod::RefFrame::compute_relative_state(), copy(), decr_left(), decr_right(), incr_left(), incr_right(), initialize(), and negate().

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

- [ref_frame_state.hh](#)
- [ref_frame_state_inline.hh](#)
- [ref_frame_state.cc](#)

8.13 jeod::RefFrameTrans Class Reference

Represent the translational aspects of a reference frame's state.

```
#include <ref_frame_state.hh>
```

Public Member Functions

- [RefFrameTrans](#) (void)
Default constructor; initializes state to a null translation.
- [RefFrameTrans](#) (const [RefFrameTrans](#) &source)
Copy constructor; initializes state to that of the source.
- [RefFrameTrans](#) & operator= (const [RefFrameTrans](#) &source)
Assignment operator; copies state from the source.
- [~RefFrameTrans](#) (void)
Destructor; does nothing.
- void [initialize](#) ()
Initialize a [RefFrameTrans](#) to a null offset.
- void [copy](#) (const [RefFrameTrans](#) &source)
Initialize a [RefFrameTrans](#) from a source state.

Data Fields

- double [position](#) [3]
Position of the subject reference frame origin with respect to the parent frame origin and expressed in parent reference frame coordinates.
- double [velocity](#) [3]
Velocity of the subject reference frame origin with respect to the parent frame origin and expressed in parent reference frame coordinates.

Friends

- class [InputProcessor](#)
- void [init_attrjeod__RefFrameTrans](#) ()

8.13.1 Detailed Description

Represent the translational aspects of a reference frame's state.

Definition at line 80 of file `ref_frame_state.hh`.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 RefFrameTrans() [1/2]

```
jeod::RefFrameTrans::RefFrameTrans (
    void ) [inline]
```

Default constructor; initializes state to a null translation.

Definition at line 84 of file `ref_frame_state_inline.hh`.

References `initialize()`.

8.13.2.2 RefFrameTrans() [2/2]

```
jeod::RefFrameTrans::RefFrameTrans (
    const RefFrameTrans & source ) [inline]
```

Copy constructor; initializes state to that of the source.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 96 of file `ref_frame_state_inline.hh`.

References `copy()`.

8.13.2.3 ~RefFrameTrans()

```
jeod::RefFrameTrans::~~RefFrameTrans (
    void ) [inline]
```

Destructor; does nothing.

Definition at line 107 of file `ref_frame_state_inline.hh`.

8.13.3 Member Function Documentation

8.13.3.1 copy()

```
void jeod::RefFrameTrans::copy (
    const RefFrameTrans & source ) [inline]
```

Initialize a [RefFrameTrans](#) from a source state.

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 131 of file `ref_frame_state_inline.hh`.

References `position`, and `velocity`.

Referenced by `jeod::RefFrameState::copy()`, `operator=()`, and `RefFrameTrans()`.

8.13.3.2 initialize()

```
void jeod::RefFrameTrans::initialize (
    void ) [inline]
```

Initialize a [RefFrameTrans](#) to a null offset.

Definition at line 118 of file `ref_frame_state_inline.hh`.

References `position`, and `velocity`.

Referenced by `jeod::RefFrameState::initialize()`, and `RefFrameTrans()`.

8.13.3.3 operator=()

```
RefFrameTrans & jeod::RefFrameTrans::operator= (
    const RefFrameTrans & source )
```

Assignment operator; copies state from the source.

Returns

Pointer to this

Parameters

in	<i>source</i>	Source state
----	---------------	--------------

Definition at line 126 of file ref_frame_state.cc.

References copy().

8.13.4 Friends And Related Function Documentation

8.13.4.1 init_attrjeod__RefFrameTrans

```
void init_attrjeod__RefFrameTrans ( ) [friend]
```

8.13.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 82 of file ref_frame_state.hh.

8.13.5 Field Documentation

8.13.5.1 position

```
double jeod::RefFrameTrans::position[3]
```

Position of the subject reference frame origin with respect to the parent frame origin and expressed in parent reference frame coordinates.

trick_units(m)

Definition at line 92 of file ref_frame_state.hh.

Referenced by jeod::RefFrame::compute_position_from(), jeod::RefFrame::compute_relative_state(), copy(), jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_left(), jeod::RefFrameState::incr_right(), initialize(), and jeod::RefFrameState::negate().

8.13.5.2 velocity

```
double jeod::RefFrameTrans::velocity[3]
```

Velocity of the subject reference frame origin with respect to the parent frame origin and expressed in parent reference frame coordinates.

trick_units(m/s)

Definition at line 98 of file ref_frame_state.hh.

Referenced by jeod::RefFrame::compute_relative_state(), copy(), jeod::RefFrameState::decr_left(), jeod::RefFrameState::decr_right(), jeod::RefFrameState::incr_left(), jeod::RefFrameState::incr_right(), initialize(), and jeod::RefFrameState::negate().

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

- [ref_frame_state.hh](#)
- [ref_frame_state_inline.hh](#)
- [ref_frame_state.cc](#)

8.14 jeod::SubscribeInterface Class Reference

A class that inherits from the [SubscribeInterface](#) class must provide subscribe and unsubscribe methods.

```
#include <subscription.hh>
```

Public Member Functions

- [SubscribeInterface](#) ()

Default constructor.

- virtual [~SubscribeInterface](#) ()

Destructor.

- virtual void [subscribe](#) (void)=0

Add a subscription to the object.

- virtual void [unsubscribe](#) (void)=0

Remove a subscription from the object.

8.14.1 Detailed Description

A class that inherits from the [SubscribeInterface](#) class must provide subscribe and unsubscribe methods.

Definition at line 114 of file subscription.hh.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 SubscribeInterface()

```
jeod::SubscribeInterface::SubscribeInterface ( ) [inline]
```

Default constructor.

Definition at line 124 of file subscription.hh.

8.14.2.2 ~SubscribeInterface()

```
virtual jeod::SubscribeInterface::~~SubscribeInterface ( ) [inline], [virtual]
```

Destructor.

Definition at line 129 of file subscription.hh.

8.14.3 Member Function Documentation

8.14.3.1 unsubscribe()

```
virtual void jeod::SubscribeInterface::unsubscribe (
    void ) [pure virtual]
```

Remove a subscription from the object.

8.14.3.2 subscribe()

```
virtual void jeod::SubscribeInterface::subscribe (
    void ) [pure virtual]
```

Add a subscription to the object.

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

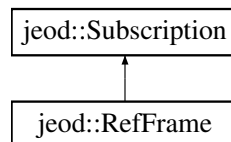
- [subscription.hh](#)

8.15 jeod::Subscription Class Reference

A [Subscription](#) object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods.

```
#include <subscription.hh>
```

Inheritance diagram for jeod::Subscription:



Public Types

- enum [Mode](#) { [Detect](#) = 0, [Subscribe](#) = 1, [Activate](#) = 2, [Freeform](#) = 3 }
- The [Subscription::Mode](#) enum specifies the mode in which a [Subscription](#) object is operating.

Public Member Functions

- [Subscription](#) ()
Subscription class default constructor.
- [Subscription](#) (Mode)
Subscription class non-default constructor.
- virtual [~Subscription](#) ()
Subscription class destructor.
- bool [is_active](#) (void) const
Return the value of the active data member.
- unsigned int [subscriptions](#) (void) const
Return the value of the subscribers data member.
- [Mode](#) [get_subscription_mode](#) (void) const
Return the value of the mode data member.
- void [activate](#) (void)
Activate a Subscription object.
- void [deactivate](#) (void)
Deactivate a Subscription object.
- void [subscribe](#) (void)
Add a subscription to a Subscription object.
- void [unsubscribe](#) (void)
Remove a subscription to a Subscription object.

Protected Member Functions

- virtual void [set_subscription_mode](#) (Mode value)
Set the value of the mode data member.
- virtual void [set_active_status](#) (bool value)
Set the active data member to the provided value.

Protected Attributes

- [Mode mode](#)
The mode in which the object is operating.
- unsigned int [subscribers](#)
Number of subscribers for this object.
- bool [active](#)
Flag indicating whether the object is active.

Friends

- class [InputProcessor](#)
- void [init_attrjeod__Subscription](#) ()

8.15.1 Detailed Description

A [Subscription](#) object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods.

The class also provides a mean for selecting only one of these two approaches as valid.

This class uses the non-virtual interface design pattern. Derived classes should not override the non-virtual public interfaces. They should instead override the private `set_active_state` method.

Definition at line 155 of file `subscription.hh`.

8.15.2 Member Enumeration Documentation

8.15.2.1 Mode

```
enum jeod::Subscription::Mode
```

The [Subscription::Mode](#) enum specifies the mode in which a [Subscription](#) object is operating.

Enumerator

Detect	First scheme used wins.
Subscribe	Activation is via subscribe/unsubscribe only.
Activate	Activation is via activate/deactivate only.
Freeform	Users can use either scheme; conflicts may arise.

Definition at line 166 of file `subscription.hh`.

8.15.3 Constructor & Destructor Documentation

8.15.3.1 Subscription() [1/2]

```
jeod::Subscription::Subscription (  
    void ) [inline]
```

[Subscription](#) class default constructor.

Definition at line 263 of file subscription.hh.

8.15.3.2 Subscription() [2/2]

```
jeod::Subscription::Subscription (  
    Mode init_mode ) [inline], [explicit]
```

[Subscription](#) class non-default constructor.

Parameters

in	<i>init_mode</i>	Initial mode
----	------------------	--------------

Definition at line 279 of file subscription.hh.

8.15.3.3 ~Subscription()

```
jeod::Subscription::~~Subscription (  
    void ) [inline], [virtual]
```

[Subscription](#) class destructor.

There are no resources to destruct.

Definition at line 294 of file subscription.hh.

8.15.4 Member Function Documentation

8.15.4.1 activate()

```
void jeod::Subscription::activate (  
    void )
```

Activate a [Subscription](#) object.

Assumptions and Limitations

- Activation is valid for this object.

Definition at line 39 of file subscription.cc.

References [Activate](#), [active](#), [Detect](#), [mode](#), [set_active_status\(\)](#), [Subscribe](#), and [jeod::RefFrameMessages↵
::subscription_error](#).

8.15.4.2 deactivate()

```
void jeod::Subscription::deactivate (  
    void )
```

Deactivate a [Subscription](#) object.

Assumptions and Limitations

- Activation is valid for this object.

Definition at line 69 of file subscription.cc.

References [Activate](#), [active](#), [Detect](#), [mode](#), [set_active_status\(\)](#), [Subscribe](#), and [jeod::RefFrameMessages↵
::subscription_error](#).

8.15.4.3 get_subscription_mode()

```
Subscription::Mode jeod::Subscription::get_subscription_mode (  
    void ) const [inline]
```

Return the value of the mode data member.

Returns

Operating mode.

Definition at line 345 of file subscription.hh.

References [mode](#).

8.15.4.4 is_active()

```
bool jeod::Subscription::is_active (
    void ) const [inline]
```

Return the value of the active data member.

Returns

Is the object active?

Definition at line 306 of file subscription.hh.

References active.

Referenced by jeod::RefFrameManager::check_ref_frame_ownership().

8.15.4.5 set_active_status()

```
void jeod::Subscription::set_active_status (
    bool value ) [protected], [virtual]
```

Set the active data member to the provided value.

Parameters

in	value	New active value
----	-------	------------------

Reimplemented in [jeod::RefFrame](#).

Definition at line 169 of file subscription.cc.

References active.

Referenced by activate(), deactivate(), jeod::RefFrame::set_active_status(), subscribe(), and unsubscribe().

8.15.4.6 set_subscription_mode()

```
void jeod::Subscription::set_subscription_mode (
    Mode value ) [inline], [protected], [virtual]
```

Set the value of the mode data member.

Parameters

in	value	Subscription mode
----	-------	-----------------------------------

Definition at line 333 of file subscription.hh.

References mode.

Referenced by jeod::RefFrame::RefFrame().

8.15.4.7 subscribe()

```
void jeod::Subscription::subscribe (
    void )
```

Add a subscription to a [Subscription](#) object.

Assumptions and Limitations

- [Subscription](#) is valid for this object.

Definition at line 99 of file subscription.cc.

References Activate, active, Detect, mode, set_active_status(), Subscribe, subscribers, and jeod::RefFrame↔ Messages::subscription_error.

Referenced by jeod::RefFrameManager::subscribe_to_frame().

8.15.4.8 subscriptions()

```
unsigned int jeod::Subscription::subscriptions (
    void ) const [inline]
```

Return the value of the subscribers data member.

Returns

Number of subscriptions.

Definition at line 320 of file subscription.hh.

References subscribers.

Referenced by jeod::RefFrameManager::frame_is_subscribed(), and jeod::RefFrameManager::unsubscribe_to_↔ frame().

8.15.4.9 unsubscribe()

```
void jeod::Subscription::unsubscribe (
    void )
```

Remove a subscription to a [Subscription](#) object.

Assumptions and Limitations

- [Subscription](#) is valid for this object.

Definition at line 131 of file subscription.cc.

References [Activate](#), [active](#), [Detect](#), [mode](#), [set_active_status\(\)](#), [Subscribe](#), [subscribers](#), and [jeod::RefFrameMessages::subscription_error](#).

Referenced by [jeod::RefFrameManager::unsubscribe_to_frame\(\)](#).

8.15.5 Friends And Related Function Documentation

8.15.5.1 init_attrjeod__Subscription

```
void init_attrjeod__Subscription ( ) [friend]
```

8.15.5.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 157 of file subscription.hh.

8.15.6 Field Documentation

8.15.6.1 active

```
bool jeod::Subscription::active [protected]
```

Flag indicating whether the object is active.

[trick_units\(-\)](#)

Definition at line 254 of file subscription.hh.

Referenced by [activate\(\)](#), [deactivate\(\)](#), [is_active\(\)](#), [set_active_status\(\)](#), [subscribe\(\)](#), and [unsubscribe\(\)](#).

8.15.6.2 mode

```
Mode jeod::Subscription::mode [protected]
```

The mode in which the object is operating.

trick_units(—)

Definition at line 244 of file subscription.hh.

Referenced by activate(), deactivate(), get_subscription_mode(), set_subscription_mode(), subscribe(), and unsubscribe().

8.15.6.3 subscribers

```
unsigned int jeod::Subscription::subscribers [protected]
```

Number of subscribers for this object.

trick_units(—)

Definition at line 249 of file subscription.hh.

Referenced by subscribe(), subscriptions(), and unsubscribe().

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

- [subscription.hh](#)
- [subscription.cc](#)

8.16 jeod::TreeLinks< Links, Container, Messages > Class Template Reference

Encapsulates links (parent, children, siblings) between objects, in the form of a tree.

```
#include <tree_links.hh>
```

Public Member Functions

- [TreeLinks](#) (Container &container_in, unsigned int path_size)
Non-default constructor.
- virtual [~TreeLinks](#) ()=default
Destructor.
- [TreeLinks](#) ()=delete
- [TreeLinks](#) (const [TreeLinks](#) &)=delete
- [TreeLinks](#) & operator= (const [TreeLinks](#) &)=delete
- Links * [child_head](#) ()
Iterator that points to the first child.
- Links * [child_tail](#) ()
Iterator that points to the last child.
- bool [is_atomic](#) ()
Is the body atomic – in other words, is it a leaf node?
- bool [has_children](#) ()
Is the body non-atomic – in other words, does it have children?
- bool [is_root](#) ()
Is the body a root node?
- Container & [container](#) ()
Accessor for the container, non-const version.
- const Container & [container](#) () const
Accessor for the container, const version.
- Links * [links_parent](#) ()
Accessor for the parent links, non-const version.
- const Links * [links_parent](#) () const
Accessor for the parent links, const version.
- Container * [parent](#) ()
Accessor for the parent container, non-const version.
- const Container * [parent](#) () const
Accessor for the parent container, const version.
- Links * [links_root](#) ()
Accessor for the root links object, non-const version.
- const Links * [links_root](#) () const
Accessor for the root links object, const version.
- Container * [root](#) ()
Accessor for the root container object, non-const version.
- const Container * [root](#) () const
Accessor for the root container object, const version.
- unsigned int [path_length](#) () const
Return the length of the path_to_node_ vector.
- unsigned int [find_path_index](#) (const Links &link) const
Find the index of the specified link in the path_to_node_.
- Container * [nth_from_root](#) (unsigned int index)
Accessor for the nth_from_root frame, non-const version.
- const Container * [nth_from_root](#) (unsigned int index) const
Accessor for the nth_from_root frame, const version.
- void [make_root](#) ()
Make the links object a root object.
- void [attach](#) (Links &new_parent)
Add this object as a child of the frame containing these links.

- void [detach](#) ()
Detach a node from its parent.
- void [reattach](#) (Links &new_parent)
Attach a node somewhere else.
- bool [is_progeny_of](#) (const Links &target) const
Determine if a node is the progeny of another.
- int [find_last_common_index](#) (const Links &target) const
Find the index of the node that represents the point of departure in the tree containing two nodes.
- const Links * [find_last_common_node](#) (const Links &target) const
Find the node that represents the point of departure in the tree containing two nodes.

Protected Member Functions

- void [construct_path_to_node](#) ()
Recursively construct the path_to_node.

Private Member Functions

- void [attach_internal](#) (Links &new_parent)
Add a frame as a child of the frame containing these links.
- void [detach_internal](#) ()
Detach a node from its parent.
- void [set_path_size](#) (unsigned int new_size)
Ensures the path size is at least as large as specified, resizing the path_to_node array if needed.

Private Attributes

- Container & [container_](#)
The object to which this set of links pertains; the container.
- Links * [parent_](#)
The [TreeLinks](#) object that is the immediate parent of this [TreeLinks](#) object in the directed tree that contains this [TreeLinks](#) object.
- std::vector< Links * > [children_](#)
The [TreeLinks](#) object's children.
- std::vector< Links * > [path_to_node_](#)
Vector of pointers to [TreeLinks](#) nodes containing the sequence of links from the root node of the tree to this [TreeLinks](#) object.

Friends

- class [InputProcessor](#)
- template<class RLinks >
class [TreeLinksAscendRange](#)
- template<class RLinks >
class [TreeLinksDescentRange](#)
- template<class RLinks >
class [TreeLinksChildrenRange](#)
- void [init_attrjeod__TreeLinks](#) ()

8.16.1 Detailed Description

```
template<class Links, class Container, class Messages>  
class jeod::TreeLinks< Links, Container, Messages >
```

Encapsulates links (parent, children, siblings) between objects, in the form of a tree.

Template Parameters

<i>Links</i>	The class being template-instantiated.
<i>Container</i>	The class that contains a TreeLinks object.
<i>Messages</i>	A message class; must contain a <code>invalid_node</code> element. This class must inherit from TreeLinks .
	Usage

This template class is designed for use with the "curiously recurring template pattern". The template parameter `Links` must be a class that derives from [TreeLinks](#): `class DerivedClass : public TreeLinks<DerivedClass, Container, Messages>`

Definition at line 100 of file `tree_links.hh`.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 `TreeLinks()` [1/3]

```
template<class Links, class Container, class Messages>
jeod::TreeLinks< Links, Container, Messages >::TreeLinks (
    Container & container_in,
    unsigned int path_size ) [inline]
```

Non-default constructor.

Parameters

<code>in, out</code>	<i>container</i> ↔ <i>_in</i>	Object that contains this object
<code>in</code>	<i>path_size</i>	Initial size to reserve for the path

Definition at line 119 of file `tree_links.hh`.

8.16.2.2 `~TreeLinks()`

```
template<class Links, class Container, class Messages>
virtual jeod::TreeLinks< Links, Container, Messages >::~~TreeLinks ( ) [virtual], [default]
```

Destructor.

8.16.2.3 TreeLinks() [2/3]

```
template<class Links, class Container, class Messages>
jeod::TreeLinks< Links, Container, Messages >::TreeLinks ( ) [delete]
```

8.16.2.4 TreeLinks() [3/3]

```
template<class Links, class Container, class Messages>
jeod::TreeLinks< Links, Container, Messages >::TreeLinks (
    const TreeLinks< Links, Container, Messages > & ) [delete]
```

8.16.3 Member Function Documentation**8.16.3.1 attach()**

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::attach (
    Links & new_parent ) [inline]
```

Add this object as a child of the frame containing these links.

This object must have no parent, no siblings.

Parameters

<i>new_parent</i>	Links object that is to be the parent of this object.
-------------------	---

Definition at line 352 of file tree_links.hh.

Referenced by jeod::RefFrame::add_child().

8.16.3.2 attach_internal()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::attach_internal (
    Links & new_parent ) [inline], [private]
```

Add a frame as a child of the frame containing these links.

Parameters

<i>new_parent</i>	The node to which this object is to be attached.
-------------------	--

Definition at line 527 of file tree_links.hh.

Referenced by jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::attach(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::reattach().

8.16.3.3 child_head()

```
template<class Links, class Container, class Messages>
Links* jeod::TreeLinks< Links, Container, Messages >::child_head ( ) [inline]
```

Iterator that points to the first child.

Definition at line 149 of file tree_links.hh.

8.16.3.4 child_tail()

```
template<class Links, class Container, class Messages>
Links* jeod::TreeLinks< Links, Container, Messages >::child_tail ( ) [inline]
```

Iterator that points to the last child.

Definition at line 157 of file tree_links.hh.

Referenced by jeod::RefFrame::~~RefFrame().

8.16.3.5 construct_path_to_node()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::construct_path_to_node ( ) [inline],
[protected]
```

Recursively construct the path_to_node.

Definition at line 495 of file tree_links.hh.

Referenced by jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::attach(), jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::detach(), jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::make_root(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::reattach().

8.16.3.6 container() [1/2]

```
template<class Links, class Container, class Messages>
Container& jeod::TreeLinks< Links, Container, Messages >::container ( ) [inline]
```

Accessor for the container, non-const version.

Returns

Object that contains this object.

Definition at line 196 of file tree_links.hh.

Referenced by jeod::RefFrame::find_last_common_node().

8.16.3.7 container() [2/2]

```
template<class Links, class Container, class Messages>
const Container& jeod::TreeLinks< Links, Container, Messages >::container ( ) const [inline]
```

Accessor for the container, const version.

Returns

Object that contains this object.

Definition at line 205 of file tree_links.hh.

8.16.3.8 detach()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::detach ( ) [inline]
```

Detach a node from its parent.

Definition at line 383 of file tree_links.hh.

Referenced by jeod::RefFrame::remove_from_parent(), and jeod::RefFrame::~RefFrame().

8.16.3.9 detach_internal()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::detach_internal ( ) [inline], [private]
```

Detach a node from its parent.

Definition at line 540 of file tree_links.hh.

Referenced by jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::detach(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::reattach().

8.16.3.10 find_last_common_index()

```
template<class Links, class Container, class Messages>
int jeod::TreeLinks< Links, Container, Messages >::find_last_common_index (
    const Links & target ) const [inline]
```

Find the index of the node that represents the point of departure in the tree containing two nodes.

Parameters

<i>target</i>	Some other node in the tree
---------------	-----------------------------

Returns

Index of the last common node

Definition at line 438 of file tree_links.hh.

Referenced by jeod::RefFrame::find_last_common_index(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::find_last_common_node().

8.16.3.11 find_last_common_node()

```
template<class Links, class Container, class Messages>
const Links* jeod::TreeLinks< Links, Container, Messages >::find_last_common_node (
    const Links & target ) const [inline]
```

Find the node that represents the point of departure in the tree containing two nodes.

Parameters

<i>target</i>	Some other node in the tree
---------------	-----------------------------

Returns

Pointer to last common node

Definition at line 481 of file tree_links.hh.

Referenced by jeod::RefFrame::find_last_common_node().

8.16.3.12 find_path_index()

```
template<class Links, class Container, class Messages>
unsigned int jeod::TreeLinks< Links, Container, Messages >::find_path_index (
    const Links & link ) const [inline]
```

Find the index of the specified link in the path_to_node_.

Definition at line 299 of file tree_links.hh.

Referenced by jeod::RefFrame::compute_pred_rel_state(), and jeod::RefFrame::compute_state_wrt_pred().

8.16.3.13 has_children()

```
template<class Links, class Container, class Messages>
bool jeod::TreeLinks< Links, Container, Messages >::has_children ( ) [inline]
```

Is the body non-atomic – in other words, does it have children?

Returns

True if the body has children, false otherwise.

Definition at line 176 of file tree_links.hh.

Referenced by jeod::RefFrame::~~RefFrame().

8.16.3.14 is_atomic()

```
template<class Links, class Container, class Messages>
bool jeod::TreeLinks< Links, Container, Messages >::is_atomic ( ) [inline]
```

Is the body atomic – in other words, is it a leaf node?

Returns

True if the body has no children, false otherwise.

Definition at line 167 of file tree_links.hh.

8.16.3.15 is_progeny_of()

```
template<class Links, class Container, class Messages>
bool jeod::TreeLinks< Links, Container, Messages >::is_progeny_of (
    const Links & target ) const [inline]
```

Determine if a node is the progeny of another.

Parameters

<i>target</i>	Target links object
---------------	---------------------

Returns

True if target is an ancestor of this node, false otherwise.

Definition at line 417 of file tree_links.hh.

Referenced by `jeod::RefFrame::is_progeny_of()`.

8.16.3.16 `is_root()`

```
template<class Links, class Container, class Messages>
bool jeod::TreeLinks< Links, Container, Messages >::is_root ( ) [inline]
```

Is the body a root node?

Returns

True if the parent is null, false otherwise.

Definition at line 186 of file `tree_links.hh`.

8.16.3.17 `links_parent()` [1/2]

```
template<class Links, class Container, class Messages>
Links* jeod::TreeLinks< Links, Container, Messages >::links_parent ( ) [inline]
```

Accessor for the parent links, non-const version.

Returns

Pointer to this object's parent [TreeLinks](#) object.

Definition at line 215 of file `tree_links.hh`.

8.16.3.18 `links_parent()` [2/2]

```
template<class Links, class Container, class Messages>
const Links* jeod::TreeLinks< Links, Container, Messages >::links_parent ( ) const [inline]
```

Accessor for the parent links, const version.

Returns

Pointer to this object's parent [TreeLinks](#) object.

Definition at line 224 of file `tree_links.hh`.

8.16.3.19 `links_root()` [1/2]

```
template<class Links, class Container, class Messages>
Links* jeod::TreeLinks< Links, Container, Messages >::links_root ( ) [inline]
```

Accessor for the root links object, non-const version.

Returns

Root links object

Definition at line 253 of file `tree_links.hh`.

8.16.3.20 `links_root()` [2/2]

```
template<class Links, class Container, class Messages>
const Links* jeod::TreeLinks< Links, Container, Messages >::links_root ( ) const [inline]
```

Accessor for the root links object, const version.

Returns

Root links object

Definition at line 262 of file `tree_links.hh`.

8.16.3.21 `make_root()`

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::make_root (
    void ) [inline]
```

Make the links object a root object.

Definition at line 332 of file `tree_links.hh`.

Referenced by `jeod::RefFrame::make_root()`.

8.16.3.22 `nth_from_root()` [1/2]

```
template<class Links, class Container, class Messages>
Container* jeod::TreeLinks< Links, Container, Messages >::nth_from_root (
    unsigned int index ) [inline]
```

Accessor for the `nth_from_root` frame, non-const version.

Parameters

<i>index</i>	Path index (root=0)
--------------	---------------------

Returns

Nth links container

Definition at line 311 of file tree_links.hh.

Referenced by `jeod::RefFrame::compute_relative_state()`.

8.16.3.23 nth_from_root() [2/2]

```
template<class Links, class Container, class Messages>
const Container* jeod::TreeLinks< Links, Container, Messages >::nth_from_root (
    unsigned int index ) const [inline]
```

Accessor for the nth_from_root frame, const version.

Parameters

<i>index</i>	Path index (root=0)
--------------	---------------------

Returns

Nth links container

Definition at line 322 of file tree_links.hh.

8.16.3.24 operator=()

```
template<class Links, class Container, class Messages>
TreeLinks& jeod::TreeLinks< Links, Container, Messages >::operator= (
    const TreeLinks< Links, Container, Messages > & ) [delete]
```

8.16.3.25 parent() [1/2]

```
template<class Links, class Container, class Messages>
Container* jeod::TreeLinks< Links, Container, Messages >::parent ( ) [inline]
```

Accessor for the parent container, non-const version.

Returns

Pointer to this object's parent Container object.

Definition at line 234 of file tree_links.hh.

Referenced by jeod::RefFrame::get_parent().

8.16.3.26 parent() [2/2]

```
template<class Links, class Container, class Messages>
const Container* jeod::TreeLinks< Links, Container, Messages >::parent ( ) const [inline]
```

Accessor for the parent container, const version.

Returns

Pointer to this object's parent Container object.

Definition at line 243 of file tree_links.hh.

8.16.3.27 path_length()

```
template<class Links, class Container, class Messages>
unsigned int jeod::TreeLinks< Links, Container, Messages >::path_length ( ) const [inline]
```

Return the length of the path_to_node_ vector.

Definition at line 290 of file tree_links.hh.

Referenced by jeod::RefFrame::compute_position_from(), jeod::RefFrame::compute_pred_rel_state(), jeod::RefFrame::compute_state_wrt_pred(), jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::find_last_common_index(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::nth_from_root().

8.16.3.28 reattach()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::reattach (
    Links & new_parent ) [inline]
```

Attach a node somewhere else.

Parameters

<i>new_parent</i>	Links object that is to be the parent of this object.
-------------------	---

Definition at line 397 of file tree_links.hh.

Referenced by jeod::RefFrame::reset_parent(), and jeod::RefFrame::transplant_node().

8.16.3.29 root() [1/2]

```
template<class Links, class Container, class Messages>
Container* jeod::TreeLinks< Links, Container, Messages >::root ( ) [inline]
```

Accessor for the root container object, non-const version.

Returns

Root container object

Definition at line 272 of file tree_links.hh.

Referenced by jeod::RefFrame::get_root().

8.16.3.30 root() [2/2]

```
template<class Links, class Container, class Messages>
const Container* jeod::TreeLinks< Links, Container, Messages >::root ( ) const [inline]
```

Accessor for the root container object, const version.

Returns

Root container object

Definition at line 281 of file tree_links.hh.

8.16.3.31 set_path_size()

```
template<class Links, class Container, class Messages>
void jeod::TreeLinks< Links, Container, Messages >::set_path_size (
    unsigned int new_size ) [inline], [private]
```

Ensures the path size is at least as large as specified, resizing the path_to_node array if needed.

Parameters

<i>new_size</i>	Requested size
-----------------	----------------

Definition at line 559 of file tree_links.hh.

Referenced by jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::construct_path_to_node(), and jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::TreeLinks().

8.16.4 Friends And Related Function Documentation

8.16.4.1 init_attrjeod__TreeLinks

```
template<class Links, class Container, class Messages>
void init_attrjeod__TreeLinks ( ) [friend]
```

8.16.4.2 InputProcessor

```
template<class Links, class Container, class Messages>
friend class InputProcessor [friend]
```

Definition at line 102 of file tree_links.hh.

8.16.4.3 TreeLinksAscendRange

```
template<class Links, class Container, class Messages>
template<class RLinks >
friend class TreeLinksAscendRange [friend]
```

Definition at line 106 of file tree_links.hh.

8.16.4.4 TreeLinksChildrenRange

```
template<class Links, class Container, class Messages>
template<class RLinks >
friend class TreeLinksChildrenRange [friend]
```

Definition at line 108 of file tree_links.hh.

8.16.4.5 TreeLinksDescentRange

```
template<class Links, class Container, class Messages>
template<class RLinks >
friend class TreeLinksDescentRange [friend]
```

Definition at line 107 of file tree_links.hh.

8.16.5 Field Documentation

8.16.5.1 children_

```
template<class Links, class Container, class Messages>
std::vector<Links*> jeod::TreeLinks< Links, Container, Messages >::children_ [private]
```

The [TreeLinks](#) object's children.

trick_units(—)

Definition at line 584 of file tree_links.hh.

Referenced by `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::child_head()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::child_tail()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::construct_path_to_node()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::has_children()`, and `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::is_atomic()`.

8.16.5.2 container_

```
template<class Links, class Container, class Messages>
Container& jeod::TreeLinks< Links, Container, Messages >::container_ [private]
```

The object to which this set of links pertains; the container.

trick_units(—)

Definition at line 572 of file tree_links.hh.

Referenced by `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::container()`.

8.16.5.3 parent_

```
template<class Links, class Container, class Messages>
Links* jeod::TreeLinks< Links, Container, Messages >::parent_ [private]
```

The [TreeLinks](#) object that is the immediate parent of this [TreeLinks](#) object in the directed tree that contains this [TreeLinks](#) object.

This pointer is null for all root objects.`trick_units(-)`

Definition at line 579 of file `tree_links.hh`.

Referenced by `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::attach()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::attach_internal()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::construct_path_to_node()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::detach_internal()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::is_root()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::links_parent()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::make_root()`, and `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::parent()`.

8.16.5.4 path_to_node_

```
template<class Links, class Container, class Messages>
std::vector<Links*> jeod::TreeLinks< Links, Container, Messages >::path_to_node_ [private]
```

Vector of pointers to [TreeLinks](#) nodes containing the sequence of links from the root node of the tree to this [TreeLinks](#) object.

The `path_to_node_` remains empty until the links object is made viable by either a call to [attach\(\)](#) or to [make_root\(\)](#). The zeroth element of this array is the root object. The last element is this node.`trick_units(-)`

Definition at line 594 of file `tree_links.hh`.

Referenced by `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::attach()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::construct_path_to_node()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::find_last_common_index()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::find_last_common_node()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::find_path_index()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::is_progeny_of()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::links_root()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::nth_from_root()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::path_length()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::root()`, `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::set_path_size()`, and `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::TreeLinks()`.

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

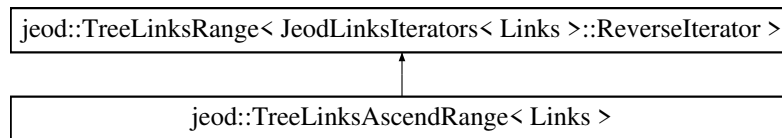
- [tree_links.hh](#)

8.17 jeod::TreeLinksAscendRange< Links > Class Template Reference

A [TreeLinksAscendRange](#) walks up a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

```
#include <tree_links.hh>
```

Inheritance diagram for jeod::TreeLinksAscendRange< Links >:



Public Types

- using [ReverseIterator](#) = typename [JeodLinksIterators](#)< Links >::ReverseIterator

Public Member Functions

- [TreeLinksAscendRange](#) (Links &links)
Non-default constructor.
- [TreeLinksAscendRange](#) (Links &links, unsigned int start_index, unsigned int end_index=0)
Non-default constructor.

8.17.1 Detailed Description

```
template<class Links>
class jeod::TreeLinksAscendRange< Links >
```

A [TreeLinksAscendRange](#) walks up a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

Definition at line 82 of file tree_links.hh.

8.17.2 Member Typedef Documentation

8.17.2.1 ReverseIterator

```
template<class Links >
using jeod::TreeLinksAscendRange< Links >::ReverseIterator = typename JeodLinksIterators<Links>↔
::ReverseIterator
```

Definition at line 176 of file tree_links_iterator.hh.

8.17.3 Constructor & Destructor Documentation

8.17.3.1 TreeLinksAscendRange() [1/2]

```
template<class Links >
jeod::TreeLinksAscendRange< Links >::TreeLinksAscendRange (
    Links & links ) [inline]
```

Non-default constructor.

Create a [TreeLinksAscendRange](#) that walks over the entire path_to_node_ from the bottom to the top.

Definition at line 183 of file tree_links_iterator.hh.

8.17.3.2 TreeLinksAscendRange() [2/2]

```
template<class Links >
jeod::TreeLinksAscendRange< Links >::TreeLinksAscendRange (
    Links & links,
    unsigned int start_index,
    unsigned int end_index = 0 ) [inline]
```

Non-default constructor.

Create a [TreeLinksAscendRange](#) given the start and end indices in the input Links object's path_to_node_ vector. Behavior is undefined if start_index > path_to_node_.size() or if end_index >= start_index.

Parameters

<i>links</i>	Object whose path_to_node_ vector is to be traversed, in reverse.
<i>start_index</i>	Index of the element in the path_to_node_ vector that immediately follows the initial element to be visited in a range-based for loop. For example, using path_to_node_.size() starts at the final element of the vector.
<i>end_index</i>	Index of the element in the path_to_node_ vector that is the last element to be visited in a range-based for loop. For example, using zero stops the iteration at the initial element in the vector.

Definition at line 208 of file tree_links_iterator.hh.

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

- [tree_links.hh](#)
- [tree_links_iterator.hh](#)

8.18 jeod::TreeLinksChildIterator< Links, Container > Class Template Reference

```
#include <class_declarations.hh>
```

8.18.1 Detailed Description

```
template<class Links, class Container>
class jeod::TreeLinksChildIterator< Links, Container >
```

Definition at line 83 of file class_declarations.hh.

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

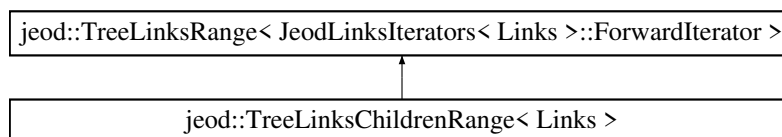
- [class_declarations.hh](#)

8.19 jeod::TreeLinksChildrenRange< Links > Class Template Reference

A [TreeLinksChildrenRange](#) walks over a Links object's children_.

```
#include <tree_links.hh>
```

Inheritance diagram for jeod::TreeLinksChildrenRange< Links >:



Public Types

- using [ForwardIterator](#) = typename [JeodLinksIterators](#)< Links >::ForwardIterator

Public Member Functions

- [TreeLinksChildrenRange](#) (Links &links)

Default constructor.

8.19.1 Detailed Description

```
template<class Links>
class jeod::TreeLinksChildrenRange< Links >
```

A [TreeLinksChildrenRange](#) walks over a Links object's children_.

Definition at line 84 of file tree_links.hh.

8.19.2 Member Typedef Documentation

8.19.2.1 ForwardIterator

```
template<class Links >
using jeod::TreeLinksChildrenRange< Links >::ForwardIterator = typename JeodLinksIterators<Links>←
::ForwardIterator
```

Definition at line 263 of file tree_links_iterator.hh.

8.19.3 Constructor & Destructor Documentation

8.19.3.1 TreeLinksChildrenRange()

```
template<class Links >
jeod::TreeLinksChildrenRange< Links >::TreeLinksChildrenRange (
    Links & links ) [inline]
```

Default constructor.

Creates a range that will visit all children.

Definition at line 269 of file tree_links_iterator.hh.

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

- [tree_links.hh](#)
- [tree_links_iterator.hh](#)

8.20 jeod::TreeLinksDescentIterator< Links, Container > Class Template Reference

```
#include <class_declarations.hh>
```

8.20.1 Detailed Description

```
template<class Links, class Container>
class jeod::TreeLinksDescentIterator< Links, Container >
```

Definition at line 82 of file class_declarations.hh.

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

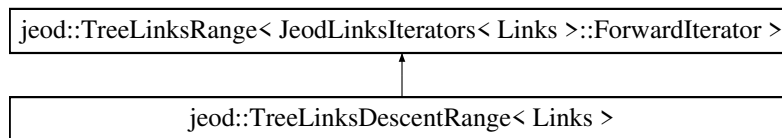
- [class_declarations.hh](#)

8.21 jeod::TreeLinksDescentRange< Links > Class Template Reference

A [TreeLinksDescentRange](#) walks down a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

```
#include <tree_links.hh>
```

Inheritance diagram for jeod::TreeLinksDescentRange< Links >:



Public Types

- using [ForwardIterator](#) = typename [JeodLinksIterators](#)< Links >::ForwardIterator

Public Member Functions

- [TreeLinksDescentRange](#) (Links &links, unsigned int start_index=0)
Constructor.

8.21.1 Detailed Description

```
template<class Links>
class jeod::TreeLinksDescentRange< Links >
```

A [TreeLinksDescentRange](#) walks down a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.

Definition at line 83 of file tree_links.hh.

8.21.2 Member Typedef Documentation

8.21.2.1 ForwardIterator

```
template<class Links >
using jeod::TreeLinksDescentRange< Links >::ForwardIterator = typename JeodLinksIterators<Links>↔
::ForwardIterator
```

Definition at line 231 of file tree_links_iterator.hh.

8.21.3 Constructor & Destructor Documentation

8.21.3.1 TreeLinksDescentRange()

```
template<class Links >
jeod::TreeLinksDescentRange< Links >::TreeLinksDescentRange (
    Links & links,
    unsigned int start_index = 0 ) [inline]
```

Constructor.

Create a [TreeLinksDescentRange](#) the marches from the start_index node of the links object's path_to_node_vector to the last node. Behavior is undefined if start_index > path_to_node_vector.size().

Parameters

<i>links</i>	Object whose path_to_node_vector is to be traversed, in reverse.
<i>start_index</i>	Index of the first node the path_to_node_vector to be visited.

Definition at line 243 of file tree_links_iterator.hh.

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

- [tree_links.hh](#)
- [tree_links_iterator.hh](#)

8.22 jeod::TreeLinksIterator< Links, Container > Class Template Reference

```
#include <class_declarations.hh>
```

8.22.1 Detailed Description

```
template<class Links, class Container>
class jeod::TreeLinksIterator< Links, Container >
```

Definition at line 80 of file class_declarations.hh.

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

- [class_declarations.hh](#)

8.23 jeod::TreeLinksParentIterator< Links, Container > Class Template Reference

```
#include <class_declarations.hh>
```

8.23.1 Detailed Description

```
template<class Links, class Container>
class jeod::TreeLinksParentIterator< Links, Container >
```

Definition at line 81 of file class_declarations.hh.

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

- [class_declarations.hh](#)

8.24 jeod::TreeLinksRange< Iterator > Class Template Reference

Base class template for all tree links range types.

```
#include <tree_links_iterator.hh>
```

Public Member Functions

- `template<typename T, typename U > TreeLinksRange (T begin_in, U end_in)`
Constructor.
- `Iterator & begin ()`
Mutable accessor to the begin_ data member.
- `Iterator & end ()`
Mutable accessor to the end_ data member.

Private Attributes

- `Iterator begin_`
Object returned (by reference) by the begin member function.
- `Iterator end_`
Object returned (by reference) by the end member function.

8.24.1 Detailed Description

```
template<class Iterator>
class jeod::TreeLinksRange< Iterator >
```

Base class template for all tree links range types.

Template Parameters

<i>Iterator</i>	The type of iterator stored as the begin_ and end_ data members and returned by the begin and end member functions.
-----------------	---

Definition at line 120 of file tree_links_iterator.hh.

8.24.2 Constructor & Destructor Documentation

8.24.2.1 TreeLinksRange()

```
template<class Iterator>
template<typename T , typename U >
jeod::TreeLinksRange< Iterator >::TreeLinksRange (
    T begin_in,
    U end_in ) [inline]
```

Constructor.

Template Parameters

<i>T</i>	The type of argument begin_in.
<i>U</i>	The type of argument end_in.

Parameters

<i>begin_↔_in</i>	Value used to construct the begin_ data member.
<i>end_in</i>	Value used to construct the end_ data member.

Definition at line 133 of file tree_links_iterator.hh.

8.24.3 Member Function Documentation

8.24.3.1 begin()

```
template<class Iterator>
Iterator& jeod::TreeLinksRange< Iterator >::begin ( ) [inline]
```

Mutable accessor to the begin_ data member.

Definition at line 143 of file tree_links_iterator.hh.

8.24.3.2 end()

```
template<class Iterator>
Iterator& jeod::TreeLinksRange< Iterator >::end ( ) [inline]
```

Mutable accessor to the end_data member.

Definition at line 148 of file tree_links_iterator.hh.

8.24.4 Field Documentation

8.24.4.1 begin_

```
template<class Iterator>
Iterator jeod::TreeLinksRange< Iterator >::begin_ [private]
```

Object returned (by reference) by the begin member function.

trick_units(-)

Definition at line 156 of file tree_links_iterator.hh.

Referenced by jeod::TreeLinksRange< JeodLinksIterators< Links >::ForwardIterator >::begin().

8.24.4.2 end_

```
template<class Iterator>
Iterator jeod::TreeLinksRange< Iterator >::end_ [private]
```

Object returned (by reference) by the end member function.

trick_units(-)

Definition at line 161 of file tree_links_iterator.hh.

Referenced by jeod::TreeLinksRange< JeodLinksIterators< Links >::ForwardIterator >::end().

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

- [tree_links_iterator.hh](#)

Chapter 9

File Documentation

9.1 base_ref_frame_manager.hh File Reference

Define the BaseRefFrameManager class, which defines the interfaces but not the implementations of the class RefFrameManager.

```
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::BaseRefFrameManager](#)
The [RefFrameManager](#) class manages the reference frames in a simulation.

Namespaces

- [jeod](#)
Namespace jeod.

9.1.1 Detailed Description

Define the BaseRefFrameManager class, which defines the interfaces but not the implementations of the class RefFrameManager.

9.2 class_declarations.hh File Reference

Forward declarations of classes defined in [ref_frame.hh](#).

Data Structures

- class [jeod::TreeLinksIterator](#)< Links, Container >
- class [jeod::TreeLinksParentIterator](#)< Links, Container >
- class [jeod::TreeLinksDescentIterator](#)< Links, Container >
- class [jeod::TreeLinksChildIterator](#)< Links, Container >

Namespaces

- [jeod](#)

Namespace jeod.

9.2.1 Detailed Description

Forward declarations of classes defined in [ref_frame.hh](#).

9.3 ref_frame.cc File Reference

Define basic methods for the RefFrame class.

```
#include <cstdint>
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/ref_frame.hh"
#include "../include/ref_frame_interface.hh"
#include "../include/tree_links_iterator.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.3.1 Detailed Description

Define basic methods for the RefFrame class.

9.4 ref_frame.hh File Reference

Define the class RefFrame.

```
#include "class_declarations.hh"
#include "subscription.hh"
#include "ref_frame_links.hh"
#include "ref_frame_state.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <string>
#include "ref_frame_inline.hh"
#include "ref_frame_interface.hh"
```

Data Structures

- class [jeod::RefFrame](#)

Describe a frame of reference and define operations on reference frames.

Namespaces

- [jeod](#)

Namespace jeod.

9.4.1 Detailed Description

Define the class RefFrame.

9.5 ref_frame_compute_relative_state.cc File Reference

Define relative state methods for the RefFrame class.

```
#include <cstdlib>
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/math/include/numerical.hh"
#include "../include/ref_frame.hh"
#include "../include/ref_frame_messages.hh"
#include "../include/ref_frame_state.hh"
#include "../include/tree_links_iterator.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.5.1 Detailed Description

Define relative state methods for the RefFrame class.

9.6 ref_frame_inline.hh File Reference

Define inline methods for the RefFrame class.

```
#include <cstdlib>
#include "ref_frame.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.6.1 Detailed Description

Define inline methods for the RefFrame class.

9.7 ref_frame_interface.hh File Reference

Define the class RefFrameOwner, which identifies an object as an "owner" of a reference frame.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "subscription.hh"
#include "class_declarations.hh"
```

Data Structures

- class [jeod::RefFrameOwner](#)
Identify an object as an "owner" of a reference frame.

Namespaces

- [jeod](#)
Namespace jeod.

9.7.1 Detailed Description

Define the class RefFrameOwner, which identifies an object as an "owner" of a reference frame.

9.8 ref_frame_items.cc File Reference

Define basic methods for the RefFrameState class.

```
#include "../include/ref_frame_items.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.8.1 Detailed Description

Define basic methods for the RefFrameState class.

9.9 ref_frame_items.hh File Reference

Define the class RefFrameItems, which identifies the aspects of a reference frame's state that have been set.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "class_declarations.hh"
#include "ref_frame_items_inline.hh"
```

Data Structures

- class [jeod::RefFrameItems](#)
Identify which aspects of a reference frame's state have been set.

Namespaces

- [jeod](#)
Namespace jeod.

9.9.1 Detailed Description

Define the class RefFrameItems, which identifies the aspects of a reference frame's state that have been set.

9.10 ref_frame_items_inline.hh File Reference

Define inline functions for the RefFrameItems::Items.

```
#include "ref_frame_items.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.10.1 Detailed Description

Define inline functions for the RefFrameItems::Items.

9.11 ref_frame_links.hh File Reference

Define the class RefFrameLinks, the class that encapsulates the links between reference frames.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "class_declarations.hh"
#include "ref_frame_messages.hh"
#include "tree_links.hh"
```

Data Structures

- class [jeod::RefFrameLinks](#)

Encapsulates the links between reference frames.

Namespaces

- [jeod](#)

Namespace jeod.

9.11.1 Detailed Description

Define the class RefFrameLinks, the class that encapsulates the links between reference frames.

MAINTENANCE NOTE – This file is, by intent, very similar to dynamics/mass/mass_body_links.hh. The version of Trick used at JEOD 2.0 beta release provided minimal support for templates. These two files should eventually be merged through the use of templates.

9.12 ref_frame_manager.cc File Reference

Define RefFrameManager methods.

```
#include <cstdint>
#include <algorithm>
#include "utils/message/include/message_handler.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/ref_frame.hh"
#include "../include/ref_frame_manager.hh"
#include "../include/ref_frame_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.12.1 Detailed Description

Define RefFrameManager methods.

9.13 ref_frame_manager.hh File Reference

Define the RefFrameManager class, which manages the reference frames in a JEOD-based simulation.

```
#include "utils/container/include/pointer_vector.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "base_ref_frame_manager.hh"
```

Data Structures

- class [jeod::RefFrameManager](#)

The [RefFrameManager](#) class manages the reference frames in a simulation.

Namespaces

- [jeod](#)

Namespace [jeod](#).

9.13.1 Detailed Description

Define the RefFrameManager class, which manages the reference frames in a JEOD-based simulation.

9.14 ref_frame_messages.cc File Reference

Implement the class RefFrameMessages.

```
#include "utils/message/include/make_message_code.hh"
#include "../include/ref_frame_messages.hh"
```

Namespaces

- [jeod](#)

Namespace [jeod](#).

Macros

- #define [MAKE_REF_FRAME_MESSAGE_CODE](#)(id) JEOD_MAKE_MESSAGE_CODE(RefFrameMessages, "utils/ref_frames/", id)

9.14.1 Detailed Description

Implement the class RefFrameMessages.

9.14.2 Macro Definition Documentation

9.14.2.1 MAKE_REF_FRAME_MESSAGE_CODE

```
#define MAKE_REF_FRAME_MESSAGE_CODE(  
    id ) JEOD_MAKE_MESSAGE_CODE(RefFrameMessages, "utils/ref_frames/", id)
```

Definition at line 37 of file ref_frame_messages.cc.

9.15 ref_frame_messages.hh File Reference

Define the class RefFrameMessages, the class that specifies the message IDs used in the reference frames model.

```
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::RefFrameMessages](#)
Declares messages associated with the reference frames model.

Namespaces

- [jeod](#)
Namespace jeod.

9.15.1 Detailed Description

Define the class RefFrameMessages, the class that specifies the message IDs used in the reference frames model.

9.16 ref_frame_set_name.cc File Reference

Define the RefFrame::set_name methods.

```
#include "../include/ref_frame.hh"  
#include "utils/named_item/include/named_item.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.16.1 Detailed Description

Define the RefFrame::set_name methods.

9.17 ref_frame_state.cc File Reference

Define methods for the RefFrameState class.

```
#include <cmath>  
#include "utils/math/include/vector3.hh"  
#include "utils/math/include/matrix3x3.hh"  
#include "utils/math/include/numerical.hh"  
#include "../include/ref_frame_state.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.17.1 Detailed Description

Define methods for the RefFrameState class.

9.18 ref_frame_state.hh File Reference

JEOD 2.0 reference frame tree class definitions.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/quaternion/include/quat.hh"
#include "class_declarations.hh"
#include "ref_frame_state_inline.hh"
```

Data Structures

- class [jeod::RefFrameTrans](#)
Represent the translational aspects of a reference frame's state.
- class [jeod::RefFrameRot](#)
Represent the rotational aspects of a reference frame's state.
- class [jeod::RefFrameState](#)
Represent a reference frame's state.

Namespaces

- [jeod](#)

Namespace jeod.

9.18.1 Detailed Description

JEOD 2.0 reference frame tree class definitions.

9.19 ref_frame_state_inline.hh File Reference

Define inline methods for the RefFrameState class and its component.

```
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "ref_frame_state.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.19.1 Detailed Description

Define inline methods for the RefFrameState class and its component.

9.20 subscription.cc File Reference

Define non-inlined methods for the Subscription class.

```
#include "utils/message/include/message_handler.hh"
#include "../include/subscription.hh"
#include "../include/ref_frame_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.20.1 Detailed Description

Define non-inlined methods for the Subscription class.

9.21 subscription.hh File Reference

Define the class Subscription.

```
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::ActivateInterface](#)

A class that inherits from the [ActivateInterface](#) class must provide activate and deactivate methods.

- class [jeod::SubscribeInterface](#)

A class that inherits from the [SubscribeInterface](#) class must provide subscribe and unsubscribe methods.

- class [jeod::Subscription](#)

A [Subscription](#) object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods.

Namespaces

- [jeod](#)

Namespace jeod.

9.21.1 Detailed Description

Define the class Subscription.

9.22 tree_links.hh File Reference

Define the template class TreeLinks, the class that encapsulates the parent/ child links between objects.

```
#include "class_declarations.hh"
#include "utils/container/include/pointer_vector.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include <algorithm>
#include <cstdlib>
#include <cstring>
#include <vector>
```

Data Structures

- class [jeod::TreeLinksAscendRange< Links >](#)
A [TreeLinksAscendRange](#) walks up a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.
- class [jeod::TreeLinksDescentRange< Links >](#)
A [TreeLinksDescentRange](#) walks down a Links object's path_to_node_ data member, starting at the start node and ending just before the end node.
- class [jeod::TreeLinksChildrenRange< Links >](#)
A [TreeLinksChildrenRange](#) walks over a Links object's children_.
- class [jeod::TreeLinks< Links, Container, Messages >](#)
Encapsulates links (parent, children, siblings) between objects, in the form of a tree.

Namespaces

- [jeod](#)

Namespace jeod.

9.22.1 Detailed Description

Define the template class TreeLinks, the class that encapsulates the parent/ child links between objects.

9.23 tree_links_iterator.hh File Reference

Define the template `TreeLinksRange` and related templates, which are used to iterate over trees.

```
#include "class_declarations.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <vector>
```

Data Structures

- struct [jeod::JeodLinksIterators< Links >](#)
Class template that defines member types `ForwardIterator` and `ReverseIterator` for walking over a `std::vector` of pointers to `Links` objects.
- struct [jeod::JeodLinksIterators< const Links >](#)
Partial specialization of [JeodLinksIterators](#) for `const Links` types.
- class [jeod::TreeLinksRange< Iterator >](#)
Base class template for all tree links range types.
- class [jeod::TreeLinksAscendRange< Links >](#)
A [TreeLinksAscendRange](#) walks up a `Links` object's `path_to_node_` data member, starting at the start node and ending just before the end node.
- class [jeod::TreeLinksDescentRange< Links >](#)
A [TreeLinksDescentRange](#) walks down a `Links` object's `path_to_node_` data member, starting at the start node and ending just before the end node.
- class [jeod::TreeLinksChildrenRange< Links >](#)
A [TreeLinksChildrenRange](#) walks over a `Links` object's `children_`.

Namespaces

- [jeod](#)
Namespace `jeod`.

9.23.1 Detailed Description

Define the template `TreeLinksRange` and related templates, which are used to iterate over trees.

The JEOD 4.0 version of the tree links iterators is motivated by the c++11 range-based for, which requires a range expression from which a begin iterator and an end sentinel can be formed.

One way the compiler can form the begin iterator and end sentinel is to have the range expression be an object that implements `begin()` and `end()` member functions. The loops that use the JEOD 4.0 tree links iterators are of the form

```
for (auto element : TreeLinksSomeRange<LinksType>(arglist)) {
    body;
}
```

Index

- ~ActivateInterface
 - jeod::ActivateInterface, [18](#)
- ~BaseRefFrameManager
 - jeod::BaseRefFrameManager, [20](#)
- ~RefFrame
 - jeod::RefFrame, [30](#)
- ~RefFrameLinks
 - jeod::RefFrameLinks, [54](#)
- ~RefFrameManager
 - jeod::RefFrameManager, [58](#)
- ~RefFrameOwner
 - jeod::RefFrameOwner, [72](#)
- ~RefFrameRot
 - jeod::RefFrameRot, [74](#)
- ~RefFrameState
 - jeod::RefFrameState, [81](#)
- ~RefFrameTrans
 - jeod::RefFrameTrans, [87](#)
- ~SubscribeInterface
 - jeod::SubscribeInterface, [90](#)
- ~Subscription
 - jeod::Subscription, [93](#)
- ~TreeLinks
 - jeod::TreeLinks, [102](#)
- activate
 - jeod::ActivateInterface, [18](#)
 - jeod::Subscription, [93](#)
- ActivateInterface
 - jeod::ActivateInterface, [17](#)
- active
 - jeod::Subscription, [97](#)
- add
 - jeod::RefFrameItems, [48](#)
- add_child
 - jeod::RefFrame, [30](#)
- add_frame_to_tree
 - jeod::BaseRefFrameManager, [20](#)
 - jeod::RefFrameManager, [58](#)
- add_ref_frame
 - jeod::BaseRefFrameManager, [20](#)
 - jeod::RefFrameManager, [59](#)
- ang_vel_mag
 - jeod::RefFrameRot, [77](#)
- ang_vel_this
 - jeod::RefFrameRot, [77](#)
- ang_vel_unit
 - jeod::RefFrameRot, [78](#)
- attach
 - jeod::TreeLinks, [103](#)
- attach_info
 - jeod::RefFrameMessages, [68](#)
- attach_internal
 - jeod::TreeLinks, [103](#)
- base_ref_frame_manager.hh, [125](#)
- begin
 - jeod::TreeLinksRange, [123](#)
- begin_
 - jeod::TreeLinksRange, [124](#)
- check_ref_frame_ownership
 - jeod::BaseRefFrameManager, [21](#)
 - jeod::RefFrameManager, [59](#)
- child_head
 - jeod::TreeLinks, [104](#)
- child_tail
 - jeod::TreeLinks, [104](#)
- children_
 - jeod::TreeLinks, [114](#)
- class_declarations.hh, [125](#)
- compute_ang_vel_products
 - jeod::RefFrameRot, [75](#)
- compute_ang_vel_unit
 - jeod::RefFrameRot, [75](#)
- compute_position_from
 - jeod::RefFrame, [31](#)
- compute_pred_rel_state
 - jeod::RefFrame, [31](#), [32](#)
- compute_quaternion
 - jeod::RefFrameRot, [75](#)
- compute_relative_state
 - jeod::RefFrame, [32](#), [33](#)
- compute_state_wrt_pred
 - jeod::RefFrame, [34](#)
- compute_transformation
 - jeod::RefFrameRot, [75](#)
- construct_path_to_node
 - jeod::TreeLinks, [104](#)
- container
 - jeod::TreeLinks, [104](#), [105](#)
- container_
 - jeod::TreeLinks, [114](#)
- contains
 - jeod::RefFrameItems, [49](#)
- copy
 - jeod::RefFrameRot, [76](#)
 - jeod::RefFrameState, [81](#)
 - jeod::RefFrameTrans, [87](#)

- deactivate
 - jeod::ActivateInterface, 18
 - jeod::Subscription, 94
- decr_left
 - jeod::RefFrameState, 81
- decr_right
 - jeod::RefFrameState, 82
- default_path_size
 - jeod::RefFrameLinks, 56
- unsubscribe
 - jeod::SubscribeInterface, 90
- detach
 - jeod::TreeLinks, 105
- detach_internal
 - jeod::TreeLinks, 105
- duplicate_entry
 - jeod::RefFrameMessages, 68
- end
 - jeod::TreeLinksRange, 123
- end_
 - jeod::TreeLinksRange, 124
- equals
 - jeod::RefFrameItems, 49
- find_last_common_index
 - jeod::RefFrame, 35
 - jeod::TreeLinks, 105
- find_last_common_node
 - jeod::RefFrame, 35
 - jeod::TreeLinks, 106
- find_path_index
 - jeod::TreeLinks, 106
- find_ref_frame
 - jeod::BaseRefFrameManager, 21
 - jeod::RefFrameManager, 59, 60
- ForwardIterator
 - jeod::JeodLinksIterators, 25
 - jeod::JeodLinksIterators< const Links >, 26
 - jeod::TreeLinksChildrenRange, 118
 - jeod::TreeLinksDescentRange, 120
- frame_is_subscribed
 - jeod::BaseRefFrameManager, 22
 - jeod::RefFrameManager, 60, 61
- get
 - jeod::RefFrameItems, 50
- get_name
 - jeod::RefFrame, 36
- get_owner
 - jeod::RefFrame, 36
- get_parent
 - jeod::RefFrame, 36
- get_root
 - jeod::RefFrame, 37
- get_subscription_mode
 - jeod::Subscription, 94
- has_children
 - jeod::TreeLinks, 106
- inconsistent_setup
 - jeod::RefFrameMessages, 68
- incr_left
 - jeod::RefFrameState, 82
- incr_right
 - jeod::RefFrameState, 82
- init_attrjeod__BaseRefFrameManager
 - jeod::BaseRefFrameManager, 24
- init_attrjeod__RefFrame
 - jeod::RefFrame, 44
- init_attrjeod__RefFrameItems
 - jeod::RefFrameItems, 52
- init_attrjeod__RefFrameLinks
 - jeod::RefFrameLinks, 55
- init_attrjeod__RefFrameManager
 - jeod::RefFrameManager, 65
- init_attrjeod__RefFrameMessages
 - jeod::RefFrameMessages, 67
- init_attrjeod__RefFrameRot
 - jeod::RefFrameRot, 77
- init_attrjeod__RefFrameState
 - jeod::RefFrameState, 84
- init_attrjeod__RefFrameTrans
 - jeod::RefFrameTrans, 88
- init_attrjeod__Subscription
 - jeod::Subscription, 97
- init_attrjeod__TreeLinks
 - jeod::TreeLinks, 113
- initialize
 - jeod::RefFrameRot, 76
 - jeod::RefFrameState, 83
 - jeod::RefFrameTrans, 87
- InputProcessor
 - jeod::BaseRefFrameManager, 24
 - jeod::RefFrame, 44
 - jeod::RefFrameItems, 52
 - jeod::RefFrameLinks, 55
 - jeod::RefFrameManager, 65
 - jeod::RefFrameMessages, 67
 - jeod::RefFrameRot, 77
 - jeod::RefFrameState, 84
 - jeod::RefFrameTrans, 88
 - jeod::Subscription, 97
 - jeod::TreeLinks, 113
- internal_error
 - jeod::RefFrameMessages, 68
- invalid_attach
 - jeod::RefFrameMessages, 69
- invalid_detach
 - jeod::RefFrameMessages, 69
- invalid_enum
 - jeod::RefFrameMessages, 69
- invalid_item
 - jeod::RefFrameMessages, 69
- invalid_name
 - jeod::RefFrameMessages, 70
- invalid_node

- jeod::RefFrameMessages, 70
- is_active
 - jeod::Subscription, 94
- is_atomic
 - jeod::TreeLinks, 107
- is_empty
 - jeod::RefFrameItems, 50
- is_full
 - jeod::RefFrameItems, 50
- is_progeny_of
 - jeod::RefFrame, 37
 - jeod::TreeLinks, 107
- is_root
 - jeod::TreeLinks, 108
- Items
 - jeod::RefFrameItems, 47
- jeod, 15
- jeod::ActivateInterface, 17
 - ~ActivateInterface, 18
 - activate, 18
 - ActivateInterface, 17
 - deactivate, 18
- jeod::BaseRefFrameManager, 19
 - ~BaseRefFrameManager, 20
 - add_frame_to_tree, 20
 - add_ref_frame, 20
 - check_ref_frame_ownership, 21
 - find_ref_frame, 21
 - frame_is_subscribed, 22
 - init_attrjeod__BaseRefFrameManager, 24
 - InputProcessor, 24
 - remove_ref_frame, 22
 - reset_tree_root_node, 23
 - subscribe_to_frame, 23
 - unsubscribe_to_frame, 23, 24
- jeod::JeodLinksIterators
 - ForwardIterator, 25
 - ReverseIterator, 25
- jeod::JeodLinksIterators< const Links >, 26
 - ForwardIterator, 26
 - ReverseIterator, 26
- jeod::JeodLinksIterators< Links >, 25
- jeod::RefFrame, 27
 - ~RefFrame, 30
 - add_child, 30
 - compute_position_from, 31
 - compute_pred_rel_state, 31, 32
 - compute_relative_state, 32, 33
 - compute_state_wrt_pred, 34
 - find_last_common_index, 35
 - find_last_common_node, 35
 - get_name, 36
 - get_owner, 36
 - get_parent, 36
 - get_root, 37
 - init_attrjeod__RefFrame, 44
 - InputProcessor, 44
 - is_progeny_of, 37
 - links, 44
 - make_root, 38
 - name, 44
 - operator=, 38
 - owner, 45
 - RefFrame, 30
 - RefFrameLinks, 44
 - remove_from_parent, 38
 - reset_parent, 38
 - set_active_status, 39
 - set_name, 39–42
 - set_owner, 42
 - set_timestamp, 43
 - state, 45
 - timestamp, 43
 - transplant_node, 43
 - update_time, 45
- jeod::RefFrameItems, 46
 - add, 48
 - contains, 49
 - equals, 49
 - get, 50
 - init_attrjeod__RefFrameItems, 52
 - InputProcessor, 52
 - is_empty, 50
 - is_full, 50
 - Items, 47
 - RefFrameItems, 48
 - remove, 50
 - set, 51
 - to_string, 51, 52
 - value, 52
- jeod::RefFrameLinks, 53
 - ~RefFrameLinks, 54
 - default_path_size, 56
 - init_attrjeod__RefFrameLinks, 55
 - InputProcessor, 55
 - operator=, 55
 - RefFrameLinks, 54, 55
- jeod::RefFrameManager, 56
 - ~RefFrameManager, 58
 - add_frame_to_tree, 58
 - add_ref_frame, 59
 - check_ref_frame_ownership, 59
 - find_ref_frame, 59, 60
 - frame_is_subscribed, 60, 61
 - init_attrjeod__RefFrameManager, 65
 - InputProcessor, 65
 - operator=, 61
 - ref_frames, 65
 - RefFrameManager, 58
 - remove_ref_frame, 61
 - reset_tree_root_node, 62
 - root_node, 65
 - subscribe_to_frame, 62
 - unsubscribe_to_frame, 63
 - validate_name, 64
- jeod::RefFrameMessages, 66

- attach_info, 68
- duplicate_entry, 68
- inconsistent_setup, 68
- init_attrjeod__RefFrameMessages, 67
- InputProcessor, 67
- internal_error, 68
- invalid_attach, 69
- invalid_detach, 69
- invalid_enum, 69
- invalid_item, 69
- invalid_name, 70
- invalid_node, 70
- null_pointer, 70
- operator=, 67
- RefFrameMessages, 67
- removal_failed, 70
- subscription_error, 71
- jeod::RefFrameOwner, 71
 - ~RefFrameOwner, 72
 - note_frame_status_change, 72
 - RefFrameOwner, 72
- jeod::RefFrameRot, 73
 - ~RefFrameRot, 74
 - ang_vel_mag, 77
 - ang_vel_this, 77
 - ang_vel_unit, 78
 - compute_ang_vel_products, 75
 - compute_ang_vel_unit, 75
 - compute_quaternion, 75
 - compute_transformation, 75
 - copy, 76
 - init_attrjeod__RefFrameRot, 77
 - initialize, 76
 - InputProcessor, 77
 - operator=, 76
 - Q_parent_this, 78
 - RefFrameRot, 74
 - T_parent_this, 78
- jeod::RefFrameState, 79
 - ~RefFrameState, 81
 - copy, 81
 - decr_left, 81
 - decr_right, 82
 - incr_left, 82
 - incr_right, 82
 - init_attrjeod__RefFrameState, 84
 - initialize, 83
 - InputProcessor, 84
 - negate, 83
 - operator=, 84
 - RefFrameState, 80
 - rot, 84
 - trans, 85
- jeod::RefFrameTrans, 85
 - ~RefFrameTrans, 87
 - copy, 87
 - init_attrjeod__RefFrameTrans, 88
 - initialize, 87
 - InputProcessor, 88
 - operator=, 87
 - position, 88
 - RefFrameTrans, 86
 - velocity, 88
- jeod::SubscribeInterface, 89
 - ~SubscribeInterface, 90
 - unsubscribe, 90
 - subscribe, 90
 - SubscribeInterface, 90
- jeod::Subscription, 91
 - ~Subscription, 93
 - activate, 93
 - active, 97
 - deactivate, 94
 - get_subscription_mode, 94
 - init_attrjeod__Subscription, 97
 - InputProcessor, 97
 - is_active, 94
 - Mode, 92
 - mode, 97
 - set_active_status, 95
 - set_subscription_mode, 95
 - subscribe, 96
 - subscribers, 98
 - Subscription, 93
 - subscriptions, 96
 - unsubscribe, 96
- jeod::TreeLinks
 - ~TreeLinks, 102
 - attach, 103
 - attach_internal, 103
 - child_head, 104
 - child_tail, 104
 - children_, 114
 - construct_path_to_node, 104
 - container, 104, 105
 - container_, 114
 - detach, 105
 - detach_internal, 105
 - find_last_common_index, 105
 - find_last_common_node, 106
 - find_path_index, 106
 - has_children, 106
 - init_attrjeod__TreeLinks, 113
 - InputProcessor, 113
 - is_atomic, 107
 - is_progeny_of, 107
 - is_root, 108
 - links_parent, 108
 - links_root, 108, 109
 - make_root, 109
 - nth_from_root, 109, 110
 - operator=, 110
 - parent, 110, 111
 - parent_, 114
 - path_length, 111
 - path_to_node_, 115

- reattach, [111](#)
- root, [112](#)
- set_path_size, [112](#)
- TreeLinks, [102](#), [103](#)
- TreeLinksAscendRange, [113](#)
- TreeLinksChildrenRange, [113](#)
- TreeLinksDescentRange, [113](#)
- jeod::TreeLinks< Links, Container, Messages >, [98](#)
- jeod::TreeLinksAscendRange
 - ReverselIterator, [116](#)
 - TreeLinksAscendRange, [117](#)
- jeod::TreeLinksAscendRange< Links >, [116](#)
- jeod::TreeLinksChildIterator< Links, Container >, [117](#)
- jeod::TreeLinksChildrenRange
 - ForwardIterator, [118](#)
 - TreeLinksChildrenRange, [119](#)
- jeod::TreeLinksChildrenRange< Links >, [118](#)
- jeod::TreeLinksDescentIterator< Links, Container >, [119](#)
- jeod::TreeLinksDescentRange
 - ForwardIterator, [120](#)
 - TreeLinksDescentRange, [121](#)
- jeod::TreeLinksDescentRange< Links >, [120](#)
- jeod::TreeLinksIterator< Links, Container >, [121](#)
- jeod::TreeLinksParentIterator< Links, Container >, [121](#)
- jeod::TreeLinksRange
 - begin, [123](#)
 - begin_, [124](#)
 - end, [123](#)
 - end_, [124](#)
 - TreeLinksRange, [123](#)
- jeod::TreeLinksRange< Iterator >, [122](#)
- links
 - jeod::RefFrame, [44](#)
- links_parent
 - jeod::TreeLinks, [108](#)
- links_root
 - jeod::TreeLinks, [108](#), [109](#)
- MAKE_REF_FRAME_MESSAGE_CODE
 - ref_frame_messages.cc, [131](#)
- make_root
 - jeod::RefFrame, [38](#)
 - jeod::TreeLinks, [109](#)
- Mode
 - jeod::Subscription, [92](#)
- mode
 - jeod::Subscription, [97](#)
- Models, [11](#)
- name
 - jeod::RefFrame, [44](#)
- negate
 - jeod::RefFrameState, [83](#)
- note_frame_status_change
 - jeod::RefFrameOwner, [72](#)
- nth_from_root
 - jeod::TreeLinks, [109](#), [110](#)
- null_pointer
 - jeod::RefFrameMessages, [70](#)
- operator=
 - jeod::RefFrame, [38](#)
 - jeod::RefFrameLinks, [55](#)
 - jeod::RefFrameManager, [61](#)
 - jeod::RefFrameMessages, [67](#)
 - jeod::RefFrameRot, [76](#)
 - jeod::RefFrameState, [84](#)
 - jeod::RefFrameTrans, [87](#)
 - jeod::TreeLinks, [110](#)
- owner
 - jeod::RefFrame, [45](#)
- parent
 - jeod::TreeLinks, [110](#), [111](#)
- parent_
 - jeod::TreeLinks, [114](#)
- path_length
 - jeod::TreeLinks, [111](#)
- path_to_node_
 - jeod::TreeLinks, [115](#)
- position
 - jeod::RefFrameTrans, [88](#)
- Q_parent_this
 - jeod::RefFrameRot, [78](#)
- reattach
 - jeod::TreeLinks, [111](#)
- ref_frame.cc, [126](#)
- ref_frame.hh, [126](#)
- ref_frame_compute_relative_state.cc, [127](#)
- ref_frame_inline.hh, [127](#)
- ref_frame_interface.hh, [128](#)
- ref_frame_items.cc, [128](#)
- ref_frame_items.hh, [129](#)
- ref_frame_items_inline.hh, [129](#)
- ref_frame_links.hh, [129](#)
- ref_frame_manager.cc, [130](#)
- ref_frame_manager.hh, [130](#)
- ref_frame_messages.cc, [131](#)
 - MAKE_REF_FRAME_MESSAGE_CODE, [131](#)
- ref_frame_messages.hh, [132](#)
- ref_frame_set_name.cc, [132](#)
- ref_frame_state.cc, [132](#)
- ref_frame_state.hh, [133](#)
- ref_frame_state_inline.hh, [133](#)
- ref_frames
 - jeod::RefFrameManager, [65](#)
- RefFrame
 - jeod::RefFrame, [30](#)
- RefFrameItems
 - jeod::RefFrameItems, [48](#)
- RefFrameLinks
 - jeod::RefFrame, [44](#)
 - jeod::RefFrameLinks, [54](#), [55](#)
- RefFrameManager

- jeod::RefFrameManager, 58
- RefFrameMessages
 - jeod::RefFrameMessages, 67
- RefFrameOwner
 - jeod::RefFrameOwner, 72
- RefFrameRot
 - jeod::RefFrameRot, 74
- RefFrameState
 - jeod::RefFrameState, 80
- RefFrameTrans
 - jeod::RefFrameTrans, 86
- RefFrames, 13
- removal_failed
 - jeod::RefFrameMessages, 70
- remove
 - jeod::RefFrameItems, 50
- remove_from_parent
 - jeod::RefFrame, 38
- remove_ref_frame
 - jeod::BaseRefFrameManager, 22
 - jeod::RefFrameManager, 61
- reset_parent
 - jeod::RefFrame, 38
- reset_tree_root_node
 - jeod::BaseRefFrameManager, 23
 - jeod::RefFrameManager, 62
- Reverseliterator
 - jeod::JeodLinksIterators, 25
 - jeod::JeodLinksIterators< const Links >, 26
 - jeod::TreeLinksAscendRange, 116
- root
 - jeod::TreeLinks, 112
- root_node
 - jeod::RefFrameManager, 65
- rot
 - jeod::RefFrameState, 84
- set
 - jeod::RefFrameItems, 51
- set_active_status
 - jeod::RefFrame, 39
 - jeod::Subscription, 95
- set_name
 - jeod::RefFrame, 39–42
- set_owner
 - jeod::RefFrame, 42
- set_path_size
 - jeod::TreeLinks, 112
- set_subscription_mode
 - jeod::Subscription, 95
- set_timestamp
 - jeod::RefFrame, 43
- state
 - jeod::RefFrame, 45
- subscribe
 - jeod::SubscribeInterface, 90
 - jeod::Subscription, 96
- subscribe_to_frame
 - jeod::BaseRefFrameManager, 23
- jeod::RefFrameManager, 62
- SubscribeInterface
 - jeod::SubscribeInterface, 90
- subscribers
 - jeod::Subscription, 98
- Subscription
 - jeod::Subscription, 93
- subscription.cc, 134
- subscription.hh, 134
- subscription_error
 - jeod::RefFrameMessages, 71
- subscriptions
 - jeod::Subscription, 96
- T_parent_this
 - jeod::RefFrameRot, 78
- timestamp
 - jeod::RefFrame, 43
- to_string
 - jeod::RefFrameItems, 51, 52
- trans
 - jeod::RefFrameState, 85
- transplant_node
 - jeod::RefFrame, 43
- tree_links.hh, 135
- tree_links_iterator.hh, 136
- TreeLinks
 - jeod::TreeLinks, 102, 103
- TreeLinksAscendRange
 - jeod::TreeLinks, 113
 - jeod::TreeLinksAscendRange, 117
- TreeLinksChildrenRange
 - jeod::TreeLinks, 113
 - jeod::TreeLinksChildrenRange, 119
- TreeLinksDescentRange
 - jeod::TreeLinks, 113
 - jeod::TreeLinksDescentRange, 121
- TreeLinksRange
 - jeod::TreeLinksRange, 123
- unsubscribe
 - jeod::Subscription, 96
- unsubscribe_to_frame
 - jeod::BaseRefFrameManager, 23, 24
 - jeod::RefFrameManager, 63
- update_time
 - jeod::RefFrame, 45
- Utils, 12
- validate_name
 - jeod::RefFrameManager, 64
- value
 - jeod::RefFrameItems, 52
- velocity
 - jeod::RefFrameTrans, 88