

# ReferenceFrameModel

5.1

Generated by Doxygen 1.8.5

Mon Jul 31 2023 11:44:37



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules . . . . .	1
<b>2</b>	<b>Namespace Index</b>	<b>3</b>
2.1	Namespace List . . . . .	3
<b>3</b>	<b>Hierarchical Index</b>	<b>5</b>
3.1	Class Hierarchy . . . . .	5
<b>4</b>	<b>Data Structure Index</b>	<b>7</b>
4.1	Data Structures . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Module Documentation</b>	<b>11</b>
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
<b>7</b>	<b>Namespace Documentation</b>	<b>15</b>
7.1	jeod Namespace Reference . . . . .	15
7.1.1	Detailed Description . . . . .	16
<b>8</b>	<b>Data Structure Documentation</b>	<b>17</b>
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 . . . . .	17
8.1.2.2	~ActivateInterface . . . . .	17
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	18
8.2.1	Detailed Description	19
8.2.2	Constructor & Destructor Documentation	19
8.2.2.1	~BaseRefFrameManager	19
8.2.3	Member Function Documentation	19
8.2.3.1	add_frame_to_tree	19
8.2.3.2	add_ref_frame	19
8.2.3.3	check_ref_frame_ownership	19
8.2.3.4	find_ref_frame	20
8.2.3.5	find_ref_frame	20
8.2.3.6	frame_is_subscribed	20
8.2.3.7	frame_is_subscribed	20
8.2.3.8	remove_ref_frame	21
8.2.3.9	reset_tree_root_node	22
8.2.3.10	subscribe_to_frame	22
8.2.3.11	subscribe_to_frame	22
8.2.3.12	unsubscribe_to_frame	22
8.2.3.13	unsubscribe_to_frame	22
8.2.4	Friends And Related Function Documentation	22
8.2.4.1	init_attrjeod__BaseRefFrameManager	22
8.2.4.2	InputProcessor	22
8.3	jeod::JeodLinksIterators< Links > Struct Template Reference	23
8.3.1	Detailed Description	23
8.3.2	Member Typedef Documentation	23
8.3.2.1	ForwardIterator	23
8.3.2.2	ReverselIterator	23
8.4	jeod::JeodLinksIterators< const Links > Struct Template Reference	23
8.4.1	Detailed Description	24
8.4.2	Member Typedef Documentation	24
8.4.2.1	ForwardIterator	24
8.4.2.2	ReverselIterator	24
8.5	jeod::RefFrame Class Reference	24
8.5.1	Detailed Description	27
8.5.2	Constructor & Destructor Documentation	27
8.5.2.1	RefFrame	27
8.5.2.2	RefFrame	27
8.5.2.3	~RefFrame	27
8.5.3	Member Function Documentation	28

8.5.3.1	<a href="#">add_child</a>	28
8.5.3.2	<a href="#">compute_position_from</a>	29
8.5.3.3	<a href="#">compute_pred_rel_state</a>	29
8.5.3.4	<a href="#">compute_pred_rel_state</a>	29
8.5.3.5	<a href="#">compute_relative_state</a>	30
8.5.3.6	<a href="#">compute_relative_state</a>	30
8.5.3.7	<a href="#">compute_state_wrt_pred</a>	31
8.5.3.8	<a href="#">compute_state_wrt_pred</a>	31
8.5.3.9	<a href="#">find_last_common_index</a>	31
8.5.3.10	<a href="#">find_last_common_node</a>	32
8.5.3.11	<a href="#">get_name</a>	32
8.5.3.12	<a href="#">get_owner</a>	32
8.5.3.13	<a href="#">get_parent</a>	33
8.5.3.14	<a href="#">get_root</a>	33
8.5.3.15	<a href="#">is_progeny_of</a>	33
8.5.3.16	<a href="#">make_root</a>	33
8.5.3.17	<a href="#">operator=</a>	33
8.5.3.18	<a href="#">remove_from_parent</a>	33
8.5.3.19	<a href="#">reset_parent</a>	34
8.5.3.20	<a href="#">set_active_status</a>	34
8.5.3.21	<a href="#">set_name</a>	34
8.5.3.22	<a href="#">set_name</a>	34
8.5.3.23	<a href="#">set_name</a>	34
8.5.3.24	<a href="#">set_name</a>	35
8.5.3.25	<a href="#">set_name</a>	35
8.5.3.26	<a href="#">set_name</a>	35
8.5.3.27	<a href="#">set_name</a>	36
8.5.3.28	<a href="#">set_owner</a>	37
8.5.3.29	<a href="#">set_timestamp</a>	37
8.5.3.30	<a href="#">timestamp</a>	37
8.5.3.31	<a href="#">transplant_node</a>	37
8.5.4	<a href="#">Friends And Related Function Documentation</a>	38
8.5.4.1	<a href="#">init_attrjeod__RefFrame</a>	38
8.5.4.2	<a href="#">InputProcessor</a>	38
8.5.4.3	<a href="#">RefFrameLinks</a>	38
8.5.5	<a href="#">Field Documentation</a>	38
8.5.5.1	<a href="#">links</a>	38
8.5.5.2	<a href="#">name</a>	38
8.5.5.3	<a href="#">owner</a>	38
8.5.5.4	<a href="#">state</a>	38

8.5.5.5	update_time	39
8.6	jeod::RefFrameItems Class Reference	39
8.6.1	Detailed Description	40
8.6.2	Member Enumeration Documentation	40
8.6.2.1	Items	40
8.6.3	Constructor & Destructor Documentation	41
8.6.3.1	RefFrameItems	41
8.6.3.2	RefFrameItems	41
8.6.4	Member Function Documentation	41
8.6.4.1	add	41
8.6.4.2	contains	41
8.6.4.3	equals	42
8.6.4.4	get	42
8.6.4.5	is_empty	42
8.6.4.6	is_full	42
8.6.4.7	remove	43
8.6.4.8	set	43
8.6.4.9	to_string	43
8.6.4.10	to_string	43
8.6.5	Friends And Related Function Documentation	44
8.6.5.1	init_attrjeod__RefFrameItems	44
8.6.5.2	InputProcessor	44
8.6.6	Field Documentation	44
8.6.6.1	value	44
8.7	jeod::RefFrameLinks Class Reference	44
8.7.1	Detailed Description	45
8.7.2	Constructor & Destructor Documentation	45
8.7.2.1	RefFrameLinks	45
8.7.2.2	~RefFrameLinks	45
8.7.2.3	RefFrameLinks	45
8.7.2.4	RefFrameLinks	45
8.7.3	Member Function Documentation	45
8.7.3.1	operator=	45
8.7.4	Friends And Related Function Documentation	46
8.7.4.1	init_attrjeod__RefFrameLinks	46
8.7.4.2	InputProcessor	46
8.7.5	Field Documentation	46
8.7.5.1	default_path_size	46
8.8	jeod::RefFrameManager Class Reference	46
8.8.1	Detailed Description	47

8.8.2	Constructor & Destructor Documentation	48
8.8.2.1	RefFrameManager	48
8.8.2.2	~RefFrameManager	48
8.8.2.3	RefFrameManager	48
8.8.3	Member Function Documentation	48
8.8.3.1	add_frame_to_tree	48
8.8.3.2	add_ref_frame	48
8.8.3.3	check_ref_frame_ownership	48
8.8.3.4	find_ref_frame	49
8.8.3.5	find_ref_frame	49
8.8.3.6	frame_is_subscribed	49
8.8.3.7	frame_is_subscribed	50
8.8.3.8	operator=	51
8.8.3.9	remove_ref_frame	51
8.8.3.10	reset_tree_root_node	51
8.8.3.11	subscribe_to_frame	51
8.8.3.12	subscribe_to_frame	52
8.8.3.13	unsubscribe_to_frame	52
8.8.3.14	unsubscribe_to_frame	52
8.8.3.15	validate_name	52
8.8.4	Friends And Related Function Documentation	53
8.8.4.1	init_attrjeod__RefFrameManager	53
8.8.4.2	InputProcessor	53
8.8.5	Field Documentation	53
8.8.5.1	ref_frames	53
8.8.5.2	root_node	53
8.9	jeod::RefFrameMessages Class Reference	53
8.9.1	Detailed Description	54
8.9.2	Constructor & Destructor Documentation	55
8.9.2.1	RefFrameMessages	55
8.9.2.2	RefFrameMessages	55
8.9.3	Member Function Documentation	55
8.9.3.1	operator=	55
8.9.4	Friends And Related Function Documentation	55
8.9.4.1	init_attrjeod__RefFrameMessages	55
8.9.4.2	InputProcessor	55
8.9.5	Field Documentation	55
8.9.5.1	attach_info	55
8.9.5.2	duplicate_entry	55
8.9.5.3	inconsistent_setup	55

8.9.5.4	<a href="#">internal_error</a>	55
8.9.5.5	<a href="#">invalid_attach</a>	55
8.9.5.6	<a href="#">invalid_detach</a>	56
8.9.5.7	<a href="#">invalid_enum</a>	56
8.9.5.8	<a href="#">invalid_item</a>	56
8.9.5.9	<a href="#">invalid_name</a>	56
8.9.5.10	<a href="#">invalid_node</a>	56
8.9.5.11	<a href="#">null_pointer</a>	56
8.9.5.12	<a href="#">removal_failed</a>	57
8.9.5.13	<a href="#">subscription_error</a>	57
8.10	<a href="#">jeod::RefFrameOwner Class Reference</a>	57
8.10.1	<a href="#">Detailed Description</a>	57
8.10.2	<a href="#">Constructor &amp; Destructor Documentation</a>	57
8.10.2.1	<a href="#">RefFrameOwner</a>	57
8.10.2.2	<a href="#">~RefFrameOwner</a>	58
8.10.3	<a href="#">Member Function Documentation</a>	58
8.10.3.1	<a href="#">note_frame_status_change</a>	58
8.11	<a href="#">jeod::RefFrameRot Class Reference</a>	58
8.11.1	<a href="#">Detailed Description</a>	59
8.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	59
8.11.2.1	<a href="#">RefFrameRot</a>	59
8.11.2.2	<a href="#">RefFrameRot</a>	59
8.11.2.3	<a href="#">~RefFrameRot</a>	59
8.11.3	<a href="#">Member Function Documentation</a>	60
8.11.3.1	<a href="#">compute_ang_vel_products</a>	60
8.11.3.2	<a href="#">compute_ang_vel_unit</a>	60
8.11.3.3	<a href="#">compute_quaternion</a>	60
8.11.3.4	<a href="#">compute_transformation</a>	60
8.11.3.5	<a href="#">copy</a>	60
8.11.3.6	<a href="#">initialize</a>	60
8.11.3.7	<a href="#">operator=</a>	61
8.11.4	<a href="#">Friends And Related Function Documentation</a>	61
8.11.4.1	<a href="#">init_attrjeod__RefFrameRot</a>	61
8.11.4.2	<a href="#">InputProcessor</a>	61
8.11.5	<a href="#">Field Documentation</a>	61
8.11.5.1	<a href="#">ang_vel_mag</a>	61
8.11.5.2	<a href="#">ang_vel_this</a>	61
8.11.5.3	<a href="#">ang_vel_unit</a>	61
8.11.5.4	<a href="#">Q_parent_this</a>	62
8.11.5.5	<a href="#">T_parent_this</a>	62



8.12	<a href="#">jeod::RefFrameState Class Reference</a>	62
8.12.1	<a href="#">Detailed Description</a>	63
8.12.2	<a href="#">Constructor &amp; Destructor Documentation</a>	63
8.12.2.1	<a href="#">RefFrameState</a>	63
8.12.2.2	<a href="#">RefFrameState</a>	63
8.12.2.3	<a href="#">~RefFrameState</a>	64
8.12.3	<a href="#">Member Function Documentation</a>	64
8.12.3.1	<a href="#">copy</a>	64
8.12.3.2	<a href="#">decr_left</a>	64
8.12.3.3	<a href="#">decr_right</a>	64
8.12.3.4	<a href="#">incr_left</a>	65
8.12.3.5	<a href="#">incr_right</a>	66
8.12.3.6	<a href="#">initialize</a>	66
8.12.3.7	<a href="#">negate</a>	66
8.12.3.8	<a href="#">operator=</a>	66
8.12.4	<a href="#">Friends And Related Function Documentation</a>	67
8.12.4.1	<a href="#">init_attrjeod__RefFrameState</a>	67
8.12.4.2	<a href="#">InputProcessor</a>	67
8.12.5	<a href="#">Field Documentation</a>	67
8.12.5.1	<a href="#">rot</a>	67
8.12.5.2	<a href="#">trans</a>	67
8.13	<a href="#">jeod::RefFrameTrans Class Reference</a>	67
8.13.1	<a href="#">Detailed Description</a>	68
8.13.2	<a href="#">Constructor &amp; Destructor Documentation</a>	68
8.13.2.1	<a href="#">RefFrameTrans</a>	68
8.13.2.2	<a href="#">RefFrameTrans</a>	68
8.13.2.3	<a href="#">~RefFrameTrans</a>	69
8.13.3	<a href="#">Member Function Documentation</a>	69
8.13.3.1	<a href="#">copy</a>	69
8.13.3.2	<a href="#">initialize</a>	69
8.13.3.3	<a href="#">operator=</a>	69
8.13.4	<a href="#">Friends And Related Function Documentation</a>	69
8.13.4.1	<a href="#">init_attrjeod__RefFrameTrans</a>	69
8.13.4.2	<a href="#">InputProcessor</a>	69
8.13.5	<a href="#">Field Documentation</a>	69
8.13.5.1	<a href="#">position</a>	70
8.13.5.2	<a href="#">velocity</a>	70
8.14	<a href="#">jeod::SubscribeInterface Class Reference</a>	70
8.14.1	<a href="#">Detailed Description</a>	70
8.14.2	<a href="#">Constructor &amp; Destructor Documentation</a>	71

8.14.2.1	<a href="#">SubscribeInterface</a>	71
8.14.2.2	<a href="#">~SubscribeInterface</a>	71
8.14.3	<a href="#">Member Function Documentation</a>	71
8.14.3.1	<a href="#">unsubscribe</a>	71
8.14.3.2	<a href="#">subscribe</a>	71
8.15	<a href="#">jeod::Subscription Class Reference</a>	71
8.15.1	<a href="#">Detailed Description</a>	72
8.15.2	<a href="#">Member Enumeration Documentation</a>	73
8.15.2.1	<a href="#">Mode</a>	73
8.15.3	<a href="#">Constructor &amp; Destructor Documentation</a>	73
8.15.3.1	<a href="#">Subscription</a>	73
8.15.3.2	<a href="#">Subscription</a>	73
8.15.3.3	<a href="#">~Subscription</a>	73
8.15.4	<a href="#">Member Function Documentation</a>	73
8.15.4.1	<a href="#">activate</a>	73
8.15.4.2	<a href="#">deactivate</a>	74
8.15.4.3	<a href="#">get_subscription_mode</a>	74
8.15.4.4	<a href="#">is_active</a>	74
8.15.4.5	<a href="#">set_active_status</a>	74
8.15.4.6	<a href="#">set_subscription_mode</a>	74
8.15.4.7	<a href="#">subscribe</a>	75
8.15.4.8	<a href="#">subscriptions</a>	75
8.15.4.9	<a href="#">unsubscribe</a>	75
8.15.5	<a href="#">Friends And Related Function Documentation</a>	75
8.15.5.1	<a href="#">init_attrjeod__Subscription</a>	75
8.15.5.2	<a href="#">InputProcessor</a>	75
8.15.6	<a href="#">Field Documentation</a>	76
8.15.6.1	<a href="#">active</a>	76
8.15.6.2	<a href="#">mode</a>	76
8.15.6.3	<a href="#">subscribers</a>	76
8.16	<a href="#">jeod::TreeLinks&lt; Links, Container, Messages &gt; Class Template Reference</a>	76
8.16.1	<a href="#">Detailed Description</a>	78
8.16.2	<a href="#">Constructor &amp; Destructor Documentation</a>	79
8.16.2.1	<a href="#">TreeLinks</a>	79
8.16.2.2	<a href="#">~TreeLinks</a>	79
8.16.2.3	<a href="#">TreeLinks</a>	79
8.16.2.4	<a href="#">TreeLinks</a>	79
8.16.3	<a href="#">Member Function Documentation</a>	79
8.16.3.1	<a href="#">attach</a>	79
8.16.3.2	<a href="#">attach_internal</a>	79

8.16.3.3	<a href="#">child_head</a>	80
8.16.3.4	<a href="#">child_tail</a>	80
8.16.3.5	<a href="#">construct_path_to_node</a>	80
8.16.3.6	<a href="#">container</a>	80
8.16.3.7	<a href="#">container</a>	80
8.16.3.8	<a href="#">detach</a>	80
8.16.3.9	<a href="#">detach_internal</a>	81
8.16.3.10	<a href="#">find_last_common_index</a>	81
8.16.3.11	<a href="#">find_last_common_node</a>	81
8.16.3.12	<a href="#">find_path_index</a>	81
8.16.3.13	<a href="#">has_children</a>	81
8.16.3.14	<a href="#">is_atomic</a>	82
8.16.3.15	<a href="#">is_progeny_of</a>	82
8.16.3.16	<a href="#">is_root</a>	82
8.16.3.17	<a href="#">links_parent</a>	82
8.16.3.18	<a href="#">links_parent</a>	83
8.16.3.19	<a href="#">links_root</a>	83
8.16.3.20	<a href="#">links_root</a>	83
8.16.3.21	<a href="#">make_root</a>	83
8.16.3.22	<a href="#">nth_from_root</a>	83
8.16.3.23	<a href="#">nth_from_root</a>	84
8.16.3.24	<a href="#">operator=</a>	85
8.16.3.25	<a href="#">parent</a>	85
8.16.3.26	<a href="#">parent</a>	85
8.16.3.27	<a href="#">path_length</a>	85
8.16.3.28	<a href="#">reattach</a>	85
8.16.3.29	<a href="#">root</a>	86
8.16.3.30	<a href="#">root</a>	86
8.16.3.31	<a href="#">set_path_size</a>	86
8.16.4	<a href="#">Friends And Related Function Documentation</a>	86
8.16.4.1	<a href="#">init_attrjeod__TreeLinks</a>	86
8.16.4.2	<a href="#">InputProcessor</a>	86
8.16.4.3	<a href="#">TreeLinksAscendRange</a>	86
8.16.4.4	<a href="#">TreeLinksChildrenRange</a>	87
8.16.4.5	<a href="#">TreeLinksDescentRange</a>	87
8.16.5	<a href="#">Field Documentation</a>	87
8.16.5.1	<a href="#">children_</a>	87
8.16.5.2	<a href="#">container_</a>	87
8.16.5.3	<a href="#">parent_</a>	87
8.16.5.4	<a href="#">path_to_node_</a>	88

8.17	<a href="#">jeod::TreeLinksAscendRange&lt; Links &gt; Class Template Reference</a>	88
8.17.1	<a href="#">Detailed Description</a>	88
8.17.2	<a href="#">Member Typedef Documentation</a>	89
8.17.2.1	<a href="#">Reverseliterator</a>	89
8.17.3	<a href="#">Constructor &amp; Destructor Documentation</a>	89
8.17.3.1	<a href="#">TreeLinksAscendRange</a>	89
8.17.3.2	<a href="#">TreeLinksAscendRange</a>	89
8.18	<a href="#">jeod::TreeLinksChildIterator&lt; Links, Container &gt; Class Template Reference</a>	89
8.18.1	<a href="#">Detailed Description</a>	89
8.19	<a href="#">jeod::TreeLinksChildrenRange&lt; Links &gt; Class Template Reference</a>	90
8.19.1	<a href="#">Detailed Description</a>	90
8.19.2	<a href="#">Member Typedef Documentation</a>	90
8.19.2.1	<a href="#">ForwardIterator</a>	90
8.19.3	<a href="#">Constructor &amp; Destructor Documentation</a>	90
8.19.3.1	<a href="#">TreeLinksChildrenRange</a>	90
8.20	<a href="#">jeod::TreeLinksDescentIterator&lt; Links, Container &gt; Class Template Reference</a>	91
8.20.1	<a href="#">Detailed Description</a>	91
8.21	<a href="#">jeod::TreeLinksDescentRange&lt; Links &gt; Class Template Reference</a>	91
8.21.1	<a href="#">Detailed Description</a>	91
8.21.2	<a href="#">Member Typedef Documentation</a>	91
8.21.2.1	<a href="#">ForwardIterator</a>	91
8.21.3	<a href="#">Constructor &amp; Destructor Documentation</a>	92
8.21.3.1	<a href="#">TreeLinksDescentRange</a>	92
8.22	<a href="#">jeod::TreeLinksIterator&lt; Links, Container &gt; Class Template Reference</a>	92
8.22.1	<a href="#">Detailed Description</a>	92
8.23	<a href="#">jeod::TreeLinksParentIterator&lt; Links, Container &gt; Class Template Reference</a>	92
8.23.1	<a href="#">Detailed Description</a>	92
8.24	<a href="#">jeod::TreeLinksRange&lt; Iterator &gt; Class Template Reference</a>	92
8.24.1	<a href="#">Detailed Description</a>	93
8.24.2	<a href="#">Constructor &amp; Destructor Documentation</a>	93
8.24.2.1	<a href="#">TreeLinksRange</a>	93
8.24.3	<a href="#">Member Function Documentation</a>	93
8.24.3.1	<a href="#">begin</a>	93
8.24.3.2	<a href="#">end</a>	94
8.24.4	<a href="#">Field Documentation</a>	94
8.24.4.1	<a href="#">begin_</a>	94
8.24.4.2	<a href="#">end_</a>	94
<b>9</b>	<b><a href="#">File Documentation</a></b>	<b>95</b>
9.1	<a href="#">base_ref_frame_manager.hh File Reference</a>	95

9.1.1	Detailed Description	95
9.2	class_declarations.hh File Reference	95
9.2.1	Detailed Description	96
9.3	ref_frame.cc File Reference	96
9.3.1	Detailed Description	96
9.4	ref_frame.hh File Reference	96
9.4.1	Detailed Description	97
9.5	ref_frame_compute_relative_state.cc File Reference	97
9.5.1	Detailed Description	97
9.6	ref_frame_inline.hh File Reference	97
9.6.1	Detailed Description	98
9.7	ref_frame_interface.hh File Reference	98
9.7.1	Detailed Description	98
9.8	ref_frame_items.cc File Reference	98
9.8.1	Detailed Description	98
9.9	ref_frame_items.hh File Reference	99
9.9.1	Detailed Description	99
9.10	ref_frame_items_inline.hh File Reference	99
9.10.1	Detailed Description	99
9.11	ref_frame_links.hh File Reference	99
9.11.1	Detailed Description	100
9.12	ref_frame_manager.cc File Reference	100
9.12.1	Detailed Description	100
9.13	ref_frame_manager.hh File Reference	100
9.13.1	Detailed Description	101
9.14	ref_frame_messages.cc File Reference	101
9.14.1	Detailed Description	101
9.14.2	Macro Definition Documentation	101
9.14.2.1	MAKE_REF_FRAME_MESSAGE_CODE	101
9.15	ref_frame_messages.hh File Reference	102
9.15.1	Detailed Description	102
9.16	ref_frame_set_name.cc File Reference	102
9.16.1	Detailed Description	102
9.17	ref_frame_state.cc File Reference	102
9.17.1	Detailed Description	103
9.18	ref_frame_state.hh File Reference	103
9.18.1	Detailed Description	103
9.19	ref_frame_state_inline.hh File Reference	103
9.19.1	Detailed Description	104
9.20	subscription.cc File Reference	104

9.20.1 Detailed Description . . . . .	104
9.21 subscription.hh File Reference . . . . .	104
9.21.1 Detailed Description . . . . .	105
9.22 tree_links.hh File Reference . . . . .	105
9.22.1 Detailed Description . . . . .	105
9.23 tree_links_iterator.hh File Reference . . . . .	105
9.23.1 Detailed Description . . . . .	106
 <b>Index</b>	 <b>107</b>

# 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:

<a href="#">jeod</a>	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 . . . . .	18
jeod::RefFrameManager . . . . .	46
jeod::JeodLinksIterators< Links > . . . . .	23
jeod::JeodLinksIterators< const Links > . . . . .	23
jeod::RefFrameItems . . . . .	39
jeod::RefFrameMessages . . . . .	53
jeod::RefFrameOwner . . . . .	57
jeod::RefFrameRot . . . . .	58
jeod::RefFrameState . . . . .	62
jeod::RefFrameTrans . . . . .	67
jeod::SubscribeInterface . . . . .	70
jeod::Subscription . . . . .	71
jeod::RefFrame . . . . .	24
jeod::TreeLinks< Links, Container, Messages > . . . . .	76
jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages > . . . . .	76
jeod::RefFrameLinks . . . . .	44
jeod::TreeLinksChildIterator< Links, Container > . . . . .	89
jeod::TreeLinksDescentIterator< Links, Container > . . . . .	91
jeod::TreeLinksIterator< Links, Container > . . . . .	92
jeod::TreeLinksParentIterator< Links, Container > . . . . .	92
jeod::TreeLinksRange< Iterator > . . . . .	92
jeod::TreeLinksRange< JeodLinksIterators< Links >::ForwardIterator > . . . . .	92
jeod::TreeLinksChildrenRange< Links > . . . . .	90
jeod::TreeLinksDescentRange< Links > . . . . .	91
jeod::TreeLinksRange< JeodLinksIterators< Links >::ReverseIterator > . . . . .	92
jeod::TreeLinksAscendRange< Links > . . . . .	88



## Chapter 4

# Data Structure Index

### 4.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">jeod::ActivateInterface</a>	A class that inherits from the <a href="#">ActivateInterface</a> class must provide activate and deactivate methods . . . . .	17
<a href="#">jeod::BaseRefFrameManager</a>	The <a href="#">RefFrameManager</a> class manages the reference frames in a simulation . . . . .	18
<a href="#">jeod::JeodLinksIterators&lt; Links &gt;</a>	Class template that defines member types ForwardIterator and ReverseIterator for walking over a std::vector of pointers to Links objects . . . . .	23
<a href="#">jeod::JeodLinksIterators&lt; const Links &gt;</a>	Partial specialization of <a href="#">JeodLinksIterators</a> for const Links types . . . . .	23
<a href="#">jeod::RefFrame</a>	Describe a frame of reference and define operations on reference frames . . . . .	24
<a href="#">jeod::RefFrameItems</a>	Identify which aspects of a reference frame's state have been set . . . . .	39
<a href="#">jeod::RefFrameLinks</a>	Encapsulates the links between reference frames . . . . .	44
<a href="#">jeod::RefFrameManager</a>	Manages the reference frames in a simulation . . . . .	46
<a href="#">jeod::RefFrameMessages</a>	Declares messages associated with the reference frames model . . . . .	53
<a href="#">jeod::RefFrameOwner</a>	Identify an object as an "owner" of a reference frame . . . . .	57
<a href="#">jeod::RefFrameRot</a>	Represent the rotational aspects of a reference frame's state . . . . .	58
<a href="#">jeod::RefFrameState</a>	Represent a reference frame's state . . . . .	62
<a href="#">jeod::RefFrameTrans</a>	Represent the translational aspects of a reference frame's state . . . . .	67
<a href="#">jeod::SubscribeInterface</a>	A class that inherits from the <a href="#">SubscribeInterface</a> class must provide subscribe and unsubscribe methods . . . . .	70
<a href="#">jeod::Subscription</a>	A <a href="#">Subscription</a> object provides two approaches of marking something as being active or inactive: The activate and deactivate methods versus the subscribe and unsubscribe methods . . . . .	71
<a href="#">jeod::TreeLinks&lt; Links, Container, Messages &gt;</a>	Encapsulates links (parent, children, siblings) between objects, in the form of a tree . . . . .	76

<a href="#">jeod::TreeLinksAscendRange&lt; Links &gt;</a>	
A <a href="#">TreeLinksAscendRange</a> walks up a Links object's path_to_node_data member, starting at the start node and ending just before the end node	88
<a href="#">jeod::TreeLinksChildIterator&lt; Links, Container &gt;</a>	89
<a href="#">jeod::TreeLinksChildrenRange&lt; Links &gt;</a>	
A <a href="#">TreeLinksChildrenRange</a> walks over a Links object's children_	90
<a href="#">jeod::TreeLinksDescentIterator&lt; Links, Container &gt;</a>	91
<a href="#">jeod::TreeLinksDescentRange&lt; Links &gt;</a>	
A <a href="#">TreeLinksDescentRange</a> walks down a Links object's path_to_node_data member, starting at the start node and ending just before the end node	91
<a href="#">jeod::TreeLinksIterator&lt; Links, Container &gt;</a>	92
<a href="#">jeod::TreeLinksParentIterator&lt; Links, Container &gt;</a>	92
<a href="#">jeod::TreeLinksRange&lt; Iterator &gt;</a>	
Base class template for all tree links range types	92

## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

<a href="#">base_ref_frame_manager.hh</a>	Define the BaseRefFrameManager class, which defines the interfaces but not the implementations of the class RefFrameManager . . . . .	95
<a href="#">class_declarations.hh</a>	Forward declarations of classes defined in <a href="#">ref_frame.hh</a> . . . . .	95
<a href="#">ref_frame.cc</a>	Define basic methods for the RefFrame class . . . . .	96
<a href="#">ref_frame.hh</a>	Define the class RefFrame . . . . .	96
<a href="#">ref_frame_compute_relative_state.cc</a>	Define relative state methods for the RefFrame class . . . . .	97
<a href="#">ref_frame_inline.hh</a>	Define inline methods for the RefFrame class . . . . .	97
<a href="#">ref_frame_interface.hh</a>	Define the class RefFrameOwner, which identifies an object as an "owner" of a reference frame . . . . .	98
<a href="#">ref_frame_items.cc</a>	Define basic methods for the RefFrameState class . . . . .	98
<a href="#">ref_frame_items.hh</a>	Define the class RefFrameItems, which identifies the aspects of a reference frame's state that have been set . . . . .	99
<a href="#">ref_frame_items_inline.hh</a>	Define inline functions for the RefFrameItems::Items . . . . .	99
<a href="#">ref_frame_links.hh</a>	Define the class RefFrameLinks, the class that encapsulates the links between reference frames . . . . .	99
<a href="#">ref_frame_manager.cc</a>	Define RefFrameManager methods . . . . .	100
<a href="#">ref_frame_manager.hh</a>	Define the RefFrameManager class, which manages the reference frames in a JEOD-based simulation . . . . .	100
<a href="#">ref_frame_messages.cc</a>	Implement the class RefFrameMessages . . . . .	101
<a href="#">ref_frame_messages.hh</a>	Define the class RefFrameMessages, the class that specifies the message IDs used in the reference frames model . . . . .	102
<a href="#">ref_frame_set_name.cc</a>	Define the RefFrame::set_name methods . . . . .	102
<a href="#">ref_frame_state.cc</a>	Define methods for the RefFrameState class . . . . .	102

<a href="#">ref_frame_state.hh</a>	
JEOD 2.0 reference frame tree class definitions . . . . .	103
<a href="#">ref_frame_state_inline.hh</a>	
Define inline methods for the RefFrameState class and its component . . . . .	103
<a href="#">subscription.cc</a>	
Define non-inlined methods for the Subscription class . . . . .	104
<a href="#">subscription.hh</a>	
Define the class Subscription . . . . .	104
<a href="#">tree_links.hh</a>	
Define the template class TreeLinks, the class that encapsulates the parent/ child links between objects . . . . .	105
<a href="#">tree_links_iterator.hh</a>	
Define the template TreeLinksRange and related templates, which are used to iterate over trees	105



## 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 [BaseRefFrameManager](#)

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

- class [TreeLinksIterator](#)
- class [TreeLinksParentIterator](#)
- class [TreeLinksDescentIterator](#)
- class [TreeLinksChildIterator](#)
- class [RefFrame](#)

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

- class [RefFrameOwner](#)

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

- 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 [RefFrameTrans](#)

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

- class [RefFrameRot](#)

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

- class [RefFrameState](#)

*Represent a reference frame's state.*

- class [ActivateInterface](#)

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

- 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 [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 [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 [TreeLinksChildrenRange](#)  
*A [TreeLinksChildrenRange](#) walks over a [Links](#) object's `children_`.*
- class [TreeLinks](#)  
*Encapsulates links (parent, children, siblings) between objects, in the form of a tree.*
- struct [JeodLinksIterators](#)  
*Class template that defines member types `ForwardIterator` and `ReverseIterator` for walking over a `std::vector` of pointers to [Links](#) objects.*
- struct [JeodLinksIterators< const Links >](#)  
*Partial specialization of [JeodLinksIterators](#) for `const Links` types.*
- 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 jeod::ActivateInterface::ActivateInterface ( ) [inline]

Default constructor.

Definition at line 89 of file subscription.hh.

##### 8.1.2.2 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 virtual void jeod::ActivateInterface::activate ( void ) [pure virtual]

Mark the object as active.

#### 8.1.3.2 virtual void jeod::ActivateInterface::deactivate ( void ) [pure virtual]

Mark the object as inactive.

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

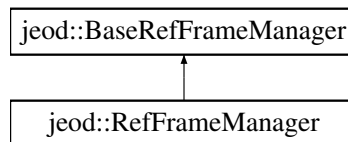
- [subscription.hh](#)

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

**8.2.4.2** `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** `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** `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 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 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:

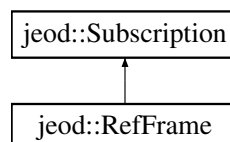
- [tree\\_links\\_iterator.hh](#)

## 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.
- [~RefFrame](#) (void) override

- 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.

  - void [set\\_active\\_status](#) (bool value) override

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

Not implemented.

#### 8.5.2.2 `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 `jeod::RefFrame::~RefFrame ( void ) [override]`

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 `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** `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** `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

<i>in</i>	<i>pred_frame</i>	The frame with respect to which the state is to be expressed
<i>out</i>	<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** `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** `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** `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 <i>wrt_frame</i> , and the transformations from this frame to the <i>wrt_frame</i> .
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** `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** `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** `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**

<i>in</i>	<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 `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**

<i>in</i>	<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 `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 `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** `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** `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** `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

<code>in</code>	<code>frame</code>	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** `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** `RefFrame& jeod::RefFrame::operator= ( const RefFrame & frame )` `[private]`

Not implemented.

**8.5.3.18** `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 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 void jeod::RefFrame::set\_active\_status ( bool value ) [override],[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 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 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 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** `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** `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** `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 `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** `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** `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** `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** `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

<code>in</code>	<code>new_parent</code>	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** `void init_attrjeod__RefFrame ( )` `[friend]`

**8.5.4.2** `friend class InputProcessor` `[friend]`

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

**8.5.4.3** `friend class RefFrameLinks` `[friend]`

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

## 8.5.5 Field Documentation

**8.5.5.1** `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** `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** `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** `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 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 `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 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 jeod::RefFrameItems::RefFrameItems ( Items new\_value ) [explicit]

Construct a [RefFrameItems](#) object.

Parameters

<i>in</i>	<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 RefFrameItems::Items jeod::RefFrameItems::add ( RefFrameItems::Items new\_items ) [inline]

Set aspects of a [RefFrameItems](#).

Returns

Updated value

Parameters

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

Definition at line 163 of file ref\_frame\_items\_inline.hh.

References `value`.

#### 8.6.4.2 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**

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

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

References value.

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

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

**Returns**

Exact equality?

**Parameters**

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

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

References value.

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

Clear aspects of a [RefFrameItems](#).

**Returns**

Updated value

**Parameters**

<i>in</i>	<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** `RefFrameItems::Items jeod::RefFrameItems::set ( RefFrameItems::Items new_value )` `[inline]`

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

**Returns**

Updated value

**Parameters**

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

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

References `value`.

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

Return a string indicating the set items.

**Returns**

Set items, by name

**Parameters**

<i>in</i>	<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** `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 void init\_attrjeod\_\_RefFrameItems ( ) [friend]

8.6.5.2 friend class InputProcessor [friend]

Definition at line 85 of file ref\_frame\_items.hh.

### 8.6.6 Field Documentation

8.6.6.1 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:

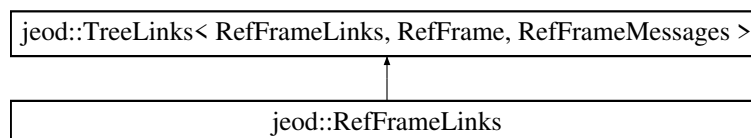
- [ref\\_frame\\_items.hh](#)
- [ref\\_frame\\_items\\_inline.hh](#)
- [ref\\_frame\\_items.cc](#)

## 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.*
- [~RefFrameLinks](#) (void) override  
*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** `jeod::RefFrameLinks::RefFrameLinks ( RefFrame & container_in )` `[inline]`, `[explicit]`

Non-default constructor.

#### Parameters

<i>container_in</i>	The <code>RefFrame</code> object that contains this object.
---------------------	---

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

**8.7.2.2** `jeod::RefFrameLinks::~~RefFrameLinks ( void )` `[inline]`, `[override]`

Destructor.

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

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

Not implemented.

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

Not implemented.

### 8.7.3 Member Function Documentation

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

Not implemented.

## 8.7.4 Friends And Related Function Documentation

8.7.4.1 `void init_attrjeod__RefFrameLinks ( ) [friend]`

8.7.4.2 `friend class InputProcessor [friend]`

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

## 8.7.5 Field Documentation

8.7.5.1 `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:

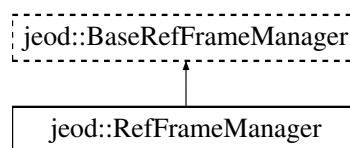
- [ref\\_frame\\_links.hh](#)

## 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.
- [~RefFrameManager](#) () override  
*RefFrameManager* destructor.
- void [add\\_ref\\_frame](#) ([RefFrame](#) &ref\_frame) override  
*Add a reference frame to the reference frame registry.*
- void [remove\\_ref\\_frame](#) ([RefFrame](#) &ref\_frame) override  
*Remove a reference frame from the reference frame registry.*
- [RefFrame](#) \* [find\\_ref\\_frame](#) (const char \*name) const override  
*Find the reference frame with the given name.*
- [RefFrame](#) \* [find\\_ref\\_frame](#) (const char \*prefix, const char \*suffix) const override  
*Find the reference frame with the dot-conjoined name "\${prefix}.\${suffix}".*
- void [check\\_ref\\_frame\\_ownership](#) (void) const override  
*Check that each active reference frame has an owner.*
- void [reset\\_tree\\_root\\_node](#) () override  
*Reset the root node in anticipation of rebuilding the entire tree.*
- void [add\\_frame\\_to\\_tree](#) ([RefFrame](#) &ref\_frame, [RefFrame](#) \*parent) override  
*Insert a reference frame in the reference frame tree.*

- void [subscribe\\_to\\_frame](#) (const char \*frame\_name) override  
*Subscribe to a reference frame, with the frame specified by name.*
- void [subscribe\\_to\\_frame](#) (RefFrame &frame) override  
*Subscribe to a reference frame, with the frame specified as an argument.*
- void [unsubscribe\\_to\\_frame](#) (const char \*frame\_name) override  
*Remove subscription to a reference frame, with the frame specified by name.*
- void [unsubscribe\\_to\\_frame](#) (RefFrame &frame) override  
*Remove subscription to a reference frame, with the frame specified as an argument.*
- bool [frame\\_is\\_subscribed](#) (const char \*frame\_name) override  
*Checks whether frame has subscribers; frame specified by name.*
- bool [frame\\_is\\_subscribed](#) (RefFrame &frame) override  
*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 `jeod::RefFrameManager::RefFrameManager ( void )`

[RefFrameManager](#) default constructor.

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

References `ref_frames`.

### 8.8.2.2 `jeod::RefFrameManager::~~RefFrameManager ( void )` `[override]`

[RefFrameManager](#) destructor.

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

References `ref_frames`.

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

Not implemented.

## 8.8.3 Member Function Documentation

### 8.8.3.1 `void jeod::RefFrameManager::add_frame_to_tree ( RefFrame & ref_frame, RefFrame * parent )` `[override]`, `[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 `void jeod::RefFrameManager::add_ref_frame ( RefFrame & ref_frame )` `[override]`, `[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 `void jeod::RefFrameManager::check_ref_frame_ownership ( void ) const` `[override]`, `[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 `RefFrame * jeod::RefFrameManager::find_ref_frame ( const char * name ) const` `[override]`, `[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 `RefFrame * jeod::RefFrameManager::find_ref_frame ( const char * prefix, const char * suffix ) const` `[override]`, `[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 `bool jeod::RefFrameManager::frame_is_subscribed ( const char * frame_name )` `[override]`, `[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** `bool jeod::RefFrameManager::frame_is_subscribed ( RefFrame & frame )` `[override],[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** `RefFrameManager& jeod::RefFrameManager::operator=( const RefFrameManager & )` `[private]`

Not implemented.

**8.8.3.9** `void jeod::RefFrameManager::remove_ref_frame ( RefFrame & ref_frame )` `[override],[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** `void jeod::RefFrameManager::reset_tree_root_node ( void )` `[override],[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** `void jeod::RefFrameManager::subscribe_to_frame ( const char * frame_name )` `[override],[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** `void jeod::RefFrameManager::subscribe_to_frame ( RefFrame & frame )` `[override],[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** `void jeod::RefFrameManager::unsubscribe_to_frame ( const char * frame_name )` `[override],[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** `void jeod::RefFrameManager::unsubscribe_to_frame ( RefFrame & frame )` `[override],[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** `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 <b>FILE</b>
<i>line</i>	Usually <b>LINE</b>
<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** `void init_attrjeod__RefFrameManager ( )` `[friend]`

**8.8.4.2** `friend class InputProcessor` `[friend]`

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

## 8.8.5 Field Documentation

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

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

## 8.9.3 Member Function Documentation

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

## 8.9.4 Friends And Related Function Documentation

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

8.9.4.2 `friend class InputProcessor` `[friend]`

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

## 8.9.5 Field Documentation

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

[RefFrameOwner](#) default constructor.

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

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

Copy constructor; initializes state to that of the source.

Parameters

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

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

References `copy()`.

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

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

#### Parameters

<code>in</code>	<code>source</code>	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 `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 RefFrameRot & jeod::RefFrameRot::operator= ( const RefFrameRot & source )

Assignment operator; copies state from the source.

##### Returns

Pointer to this

##### Parameters

<code>in</code>	<code>source</code>	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 `void init_attrjeod__RefFrameRot( ) [friend]`

8.11.4.2 `friend class InputProcessor [friend]`

Definition at line 129 of file `ref_frame_state.hh`.

### 8.11.5 Field Documentation

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

[RefFrameState](#) default constructor.

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

References `initialize()`.

#### 8.12.2.2 jeod::RefFrameState::RefFrameState ( const RefFrameState & source )

[RefFrameState](#) copy constructor.

Parameters

---

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

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

References `copy()`.

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

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

Parameters

<i>in</i>	<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 `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

<i>in</i>	<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::RefFrameRot::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 `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

<i>in</i>	<i>s_bc</i>	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::RefFrameRot::Q_parent_this`, `rot`, `jeod::RefFrameRot::T_parent_this`, `trans`, and `jeod::RefFrameTrans::velocity`.

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

#### 8.12.3.4 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

<i>in</i>	<i>s_ab</i>	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::RefFrameRot::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 `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

<i>in</i>	<i>s_bc</i>	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::RefFrameTrans::position`, `jeod::RefFrameRot::Q_parent_this`, `rot`, `jeod::RefFrameRot::T_parent_this`, `trans`, and `jeod::RefFrameTrans::velocity`.

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

Copy a reference frame state, negated.

## Parameters

<i>in</i>	<i>source</i>	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 `RefFrameState & jeod::RefFrameState::operator= ( const RefFrameState & source )`

Assignment operator; copies state from the source.



**Returns**

Pointer to this

**Parameters**

<i>in</i>	<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 `void init_attrjeod_RefFrameState ( ) [friend]`

8.12.4.2 `friend class InputProcessor [friend]`

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

**8.12.5 Field Documentation**

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

Copy constructor; initializes state to that of the source.

Parameters

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

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

References `copy()`.

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

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

Parameters

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

Assignment operator; copies state from the source.

Returns

Pointer to this

Parameters

in	source	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 void init\_attrjeod\_\_RefFrameTrans ( ) [friend]

#### 8.13.4.2 friend class InputProcessor [friend]

Definition at line 82 of file ref\_frame\_state.hh.

### 8.13.5 Field Documentation

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

Default constructor.

Definition at line 124 of file subscription.hh.

#### 8.14.2.2 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 virtual void jeod::SubscribeInterface::unsubscribe ( void ) [pure virtual]

Remove a subscription from the object.

#### 8.14.3.2 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:

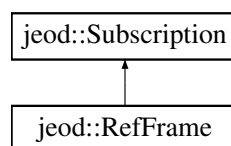
- [subscription.hh](#)

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

[Subscription](#) class default constructor.

Definition at line 263 of file subscription.hh.

### 8.15.3.2 jeod::Subscription::Subscription ( Mode init\_mode ) [inline],[explicit]

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

#### Parameters

in	init_mode	Initial mode
----	-----------	--------------

Definition at line 279 of file subscription.hh.

### 8.15.3.3 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 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 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 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 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 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 void jeod::Subscription::set\_subscription\_mode ( Mode value ) [inline],[protected],[virtual]

Set the value of the mode data member.



## Parameters

<i>in</i>	<i>value</i>	<a href="#">Subscription</a> mode
-----------	--------------	-----------------------------------

Definition at line 333 of file subscription.hh.

References mode.

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

#### 8.15.4.7 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::RefFrameMessages::subscription\_error.

Referenced by jeod::RefFrameManager::subscribe\_to\_frame().

#### 8.15.4.8 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 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 void init\_attrjeod\_\_Subscription ( ) [friend]

#### 8.15.5.2 friend class InputProcessor [friend]

Definition at line 157 of file subscription.hh.

### 8.15.6 Field Documentation

#### 8.15.6.1 `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 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 `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 <a href="#">TreeLinks</a> object.
<i>Messages</i>	A message class; must contain a invalid_node element. This class must inherit from <a href="#">TreeLinks</a> .
<b>Usage</b>	

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** `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

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

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

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

Destructor.

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

**8.16.2.4** `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** `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** `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 524 of file `tree_links.hh`.

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

**8.16.3.3** `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** `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** `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 492 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** `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** `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** `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** `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 537 of file tree\_links.hh.

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

**8.16.3.10** `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 435 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** `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 478 of file tree\_links.hh.

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

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

**8.16.3.25** `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** `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** `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** `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** `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** `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** `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 556 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** `template<class Links, class Container, class Messages> void init_attrjeod__TreeLinks ( ) [friend]`

**8.16.4.2** `template<class Links, class Container, class Messages> friend class InputProcessor [friend]`

Definition at line 102 of file tree\_links.hh.

**8.16.4.3** `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 `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 `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 `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 581 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 `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 569 of file tree\_links.hh.

Referenced by `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::container()`.

8.16.5.3 `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 576 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 `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 591 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()`, and `jeod::TreeLinks< RefFrameLinks, RefFrame, RefFrameMessages >::set_path_size()`.

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

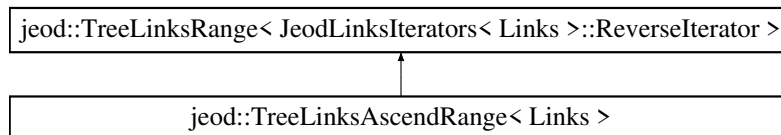
- [tree\\_links.hh](#)

## 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 `template<class Links > using jeod::TreeLinksAscendRange< Links >::Reverseliterator = typename JeodLinksIterators<Links>::Reverseliterator`

Definition at line 176 of file tree\_links\_iterator.hh.

## 8.17.3 Constructor & Destructor Documentation

8.17.3.1 `template<class Links > jeod::TreeLinksAscendRange< Links >::TreeLinksAscendRange ( Links & links ) [inline],[explicit]`

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 `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:

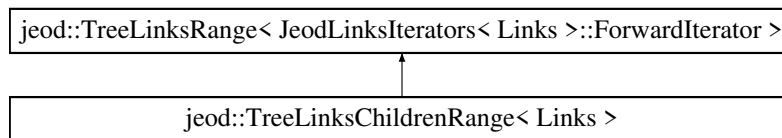
- [class\\_declarations.hh](#)

## 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 `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 `template<class Links > jeod::TreeLinksChildrenRange< Links >::TreeLinksChildrenRange ( Links &links ) [inline], [explicit]`

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:

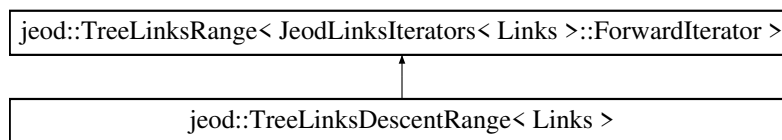
- [class\\_declarations.hh](#)

## 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 `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 `template<class Links > jeod::TreeLinksDescentRange< Links >::TreeLinksDescentRange ( Links & links, unsigned int start_index = 0 ) [inline], [explicit]`

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\_.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 `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 `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** `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** `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** `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`.

Definition in file [base\\_ref\\_frame\\_manager.hh](#).

### 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](#).

Definition in file [class\\_declarations.hh](#).

## 9.3 ref\_frame.cc File Reference

Define basic methods for the RefFrame class.

```
#include <cstdlib>
#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.

Definition in file [ref\\_frame.cc](#).

## 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.

Definition in file [ref\\_frame.hh](#).

## 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.

Definition in file [ref\\_frame\\_compute\\_relative\\_state.cc](#).

## 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.

Definition in file [ref\\_frame\\_inline.hh](#).

## 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.

Definition in file [ref\\_frame\\_interface.hh](#).

## 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.

Definition in file [ref\\_frame\\_items.cc](#).



## 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.

Definition in file [ref\\_frame\\_items.hh](#).

## 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.

Definition in file [ref\\_frame\\_items\\_inline.hh](#).

## 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.

Definition in file [ref\\_frame\\_links.hh](#).

## 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.

Definition in file [ref\\_frame\\_manager.cc](#).

## 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.

Definition in file [ref\\_frame\\_manager.hh](#).

## 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.

Definition in file [ref\\_frame\\_messages.cc](#).

### 9.14.2 Macro Definition Documentation

- 9.14.2.1 `#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.

Definition in file [ref\\_frame\\_messages.hh](#).

## 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.

Definition in file [ref\\_frame\\_set\\_name.cc](#).

## 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.

Definition in file [ref\\_frame\\_state.cc](#).

## 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.

Definition in file [ref\\_frame\\_state.hh](#).

## 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.

Definition in file [ref\\_frame\\_state\\_inline.hh](#).

## 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.

Definition in file [subscription.cc](#).

## 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.

Definition in file [subscription.hh](#).

## 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 <cstddef>
#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.

Definition in file [tree\\_links.hh](#).

## 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;
```

Definition in file [tree\\_links\\_iterator.hh](#).



# Index

- ~ActivateInterface
  - jeod::ActivateInterface, [17](#)
- ~BaseRefFrameManager
  - jeod::BaseRefFrameManager, [19](#)
- ~RefFrame
  - jeod::RefFrame, [27](#)
- ~RefFrameLinks
  - jeod::RefFrameLinks, [45](#)
- ~RefFrameManager
  - jeod::RefFrameManager, [48](#)
- ~RefFrameOwner
  - jeod::RefFrameOwner, [57](#)
- ~RefFrameRot
  - jeod::RefFrameRot, [59](#)
- ~RefFrameState
  - jeod::RefFrameState, [64](#)
- ~RefFrameTrans
  - jeod::RefFrameTrans, [68](#)
- ~SubscribeInterface
  - jeod::SubscribeInterface, [71](#)
- ~Subscription
  - jeod::Subscription, [73](#)
- ~TreeLinks
  - jeod::TreeLinks, [79](#)
- Activate
  - jeod::Subscription, [73](#)
- activate
  - jeod::ActivateInterface, [18](#)
  - jeod::Subscription, [73](#)
- ActivateInterface
  - jeod::ActivateInterface, [17](#)
- active
  - jeod::Subscription, [76](#)
- add
  - jeod::RefFrameItems, [41](#)
- add\_child
  - jeod::RefFrame, [28](#)
- add\_frame\_to\_tree
  - jeod::BaseRefFrameManager, [19](#)
  - jeod::RefFrameManager, [48](#)
- add\_ref\_frame
  - jeod::BaseRefFrameManager, [19](#)
  - jeod::RefFrameManager, [48](#)
- ang\_vel\_mag
  - jeod::RefFrameRot, [61](#)
- ang\_vel\_this
  - jeod::RefFrameRot, [61](#)
- ang\_vel\_unit
  - jeod::RefFrameRot, [61](#)

- Att
  - jeod::RefFrameItems, [40](#)
- Att\_Rate
  - jeod::RefFrameItems, [41](#)
- attach
  - jeod::TreeLinks, [79](#)
- attach\_info
  - jeod::RefFrameMessages, [55](#)
- attach\_internal
  - jeod::TreeLinks, [79](#)
- base\_ref\_frame\_manager.hh, [95](#)
- begin
  - jeod::TreeLinksRange, [93](#)
- begin\_
  - jeod::TreeLinksRange, [94](#)
- check\_ref\_frame\_ownership
  - jeod::BaseRefFrameManager, [19](#)
  - jeod::RefFrameManager, [48](#)
- child\_head
  - jeod::TreeLinks, [79](#)
- child\_tail
  - jeod::TreeLinks, [80](#)
- children\_
  - jeod::TreeLinks, [87](#)
- class\_declarations.hh, [95](#)
- compute\_ang\_vel\_products
  - jeod::RefFrameRot, [60](#)
- compute\_ang\_vel\_unit
  - jeod::RefFrameRot, [60](#)
- compute\_position\_from
  - jeod::RefFrame, [29](#)
- compute\_pred\_rel\_state
  - jeod::RefFrame, [29](#)
- compute\_quaternion
  - jeod::RefFrameRot, [60](#)
- compute\_relative\_state
  - jeod::RefFrame, [30](#)
- compute\_state\_wrt\_pred
  - jeod::RefFrame, [31](#)
- compute\_transformation
  - jeod::RefFrameRot, [60](#)
- construct\_path\_to\_node
  - jeod::TreeLinks, [80](#)
- container
  - jeod::TreeLinks, [80](#)
- container\_
  - jeod::TreeLinks, [87](#)
- contains

- jeod::RefFrameItems, 41
- copy
  - jeod::RefFrameRot, 60
  - jeod::RefFrameState, 64
  - jeod::RefFrameTrans, 69
- deactivate
  - jeod::ActivateInterface, 18
  - jeod::Subscription, 73
- decr\_left
  - jeod::RefFrameState, 64
- decr\_right
  - jeod::RefFrameState, 64
- default\_path\_size
  - jeod::RefFrameLinks, 46
- unsubscribe
  - jeod::SubscribeInterface, 71
- detach
  - jeod::TreeLinks, 80
- detach\_internal
  - jeod::TreeLinks, 80
- Detect
  - jeod::Subscription, 73
- duplicate\_entry
  - jeod::RefFrameMessages, 55
- end
  - jeod::TreeLinksRange, 93
- end\_
  - jeod::TreeLinksRange, 94
- equals
  - jeod::RefFrameItems, 42
- find\_last\_common\_index
  - jeod::RefFrame, 31
  - jeod::TreeLinks, 81
- find\_last\_common\_node
  - jeod::RefFrame, 32
  - jeod::TreeLinks, 81
- find\_path\_index
  - jeod::TreeLinks, 81
- find\_ref\_frame
  - jeod::BaseRefFrameManager, 19, 20
  - jeod::RefFrameManager, 49
- ForwardIterator
  - jeod::JeodLinksIterators, 23
  - jeod::JeodLinksIterators< const Links >, 24
  - jeod::TreeLinksChildrenRange, 90
  - jeod::TreeLinksDescentRange, 91
- frame\_is\_subscribed
  - jeod::BaseRefFrameManager, 20
  - jeod::RefFrameManager, 49
- Freeform
  - jeod::Subscription, 73
- get
  - jeod::RefFrameItems, 42
- get\_name
  - jeod::RefFrame, 32
- get\_owner
  - jeod::RefFrame, 32
- get\_parent
  - jeod::RefFrame, 32
- get\_root
  - jeod::RefFrame, 33
- get\_subscription\_mode
  - jeod::Subscription, 74
- has\_children
  - jeod::TreeLinks, 81
- inconsistent\_setup
  - jeod::RefFrameMessages, 55
- incr\_left
  - jeod::RefFrameState, 64
- incr\_right
  - jeod::RefFrameState, 66
- init\_attrjeod\_\_BaseRefFrameManager
  - jeod::BaseRefFrameManager, 22
- init\_attrjeod\_\_RefFrame
  - jeod::RefFrame, 38
- init\_attrjeod\_\_RefFrameItems
  - jeod::RefFrameItems, 44
- init\_attrjeod\_\_RefFrameLinks
  - jeod::RefFrameLinks, 46
- init\_attrjeod\_\_RefFrameManager
  - jeod::RefFrameManager, 53
- init\_attrjeod\_\_RefFrameMessages
  - jeod::RefFrameMessages, 55
- init\_attrjeod\_\_RefFrameRot
  - jeod::RefFrameRot, 61
- init\_attrjeod\_\_RefFrameState
  - jeod::RefFrameState, 67
- init\_attrjeod\_\_RefFrameTrans
  - jeod::RefFrameTrans, 69
- init\_attrjeod\_\_Subscription
  - jeod::Subscription, 75
- init\_attrjeod\_\_TreeLinks
  - jeod::TreeLinks, 86
- initialize
  - jeod::RefFrameRot, 60
  - jeod::RefFrameState, 66
  - jeod::RefFrameTrans, 69
- InputProcessor
  - jeod::BaseRefFrameManager, 22
  - jeod::RefFrame, 38
  - jeod::RefFrameItems, 44
  - jeod::RefFrameLinks, 46
  - jeod::RefFrameManager, 53
  - jeod::RefFrameMessages, 55
  - jeod::RefFrameRot, 61
  - jeod::RefFrameState, 67
  - jeod::RefFrameTrans, 69
  - jeod::Subscription, 75
  - jeod::TreeLinks, 86
- internal\_error
  - jeod::RefFrameMessages, 55
- invalid\_attach

- jeod::RefFrameMessages, 55
- invalid\_detach
  - jeod::RefFrameMessages, 56
- invalid\_enum
  - jeod::RefFrameMessages, 56
- invalid\_item
  - jeod::RefFrameMessages, 56
- invalid\_name
  - jeod::RefFrameMessages, 56
- invalid\_node
  - jeod::RefFrameMessages, 56
- is\_active
  - jeod::Subscription, 74
- is\_atomic
  - jeod::TreeLinks, 82
- is\_empty
  - jeod::RefFrameItems, 42
- is\_full
  - jeod::RefFrameItems, 42
- is\_progeny\_of
  - jeod::RefFrame, 33
  - jeod::TreeLinks, 82
- is\_root
  - jeod::TreeLinks, 82
- Items
  - jeod::RefFrameItems, 40
- jeod, 15
- jeod::RefFrameItems
  - Att, 40
  - Att\_Rate, 41
  - No\_Items, 40
  - Pos, 40
  - Pos\_Att, 40
  - Pos\_Att\_Rate, 41
  - Pos\_Rate, 41
  - Pos\_Vel, 40
  - Pos\_Vel\_Att, 40
  - Pos\_Vel\_Att\_Rate, 41
  - Pos\_Vel\_Rate, 41
  - Rate, 41
  - Vel, 40
  - Vel\_Att, 40
  - Vel\_Att\_Rate, 41
  - Vel\_Rate, 41
- jeod::Subscription
  - Activate, 73
  - Detect, 73
  - Freeform, 73
  - Subscribe, 73
- jeod::ActivateInterface, 17
  - ~ActivateInterface, 17
  - activate, 18
  - ActivateInterface, 17
  - deactivate, 18
- jeod::BaseRefFrameManager, 18
  - ~BaseRefFrameManager, 19
  - add\_frame\_to\_tree, 19
  - add\_ref\_frame, 19
  - check\_ref\_frame\_ownership, 19
  - find\_ref\_frame, 19, 20
  - frame\_is\_subscribed, 20
  - init\_attrjeod\_\_BaseRefFrameManager, 22
  - InputProcessor, 22
  - remove\_ref\_frame, 20
  - reset\_tree\_root\_node, 22
  - subscribe\_to\_frame, 22
  - unsubscribe\_to\_frame, 22
- jeod::JeodLinksIterators
  - ForwardIterator, 23
  - ReverseIterator, 23
- jeod::JeodLinksIterators< const Links >, 23
  - ForwardIterator, 24
  - ReverseIterator, 24
- jeod::JeodLinksIterators< Links >, 23
- jeod::RefFrame, 24
  - ~RefFrame, 27
  - add\_child, 28
  - compute\_position\_from, 29
  - compute\_pred\_rel\_state, 29
  - compute\_relative\_state, 30
  - compute\_state\_wrt\_pred, 31
  - find\_last\_common\_index, 31
  - find\_last\_common\_node, 32
  - get\_name, 32
  - get\_owner, 32
  - get\_parent, 32
  - get\_root, 33
  - init\_attrjeod\_\_RefFrame, 38
  - InputProcessor, 38
  - is\_progeny\_of, 33
  - links, 38
  - make\_root, 33
  - name, 38
  - operator=, 33
  - owner, 38
  - RefFrame, 27
  - RefFrameLinks, 38
  - remove\_from\_parent, 33
  - reset\_parent, 33
  - set\_active\_status, 34
  - set\_name, 34, 35
  - set\_owner, 37
  - set\_timestamp, 37
  - state, 38
  - timestamp, 37
  - transplant\_node, 37
  - update\_time, 39
- jeod::RefFrameItems, 39
  - add, 41
  - contains, 41
  - equals, 42
  - get, 42
  - init\_attrjeod\_\_RefFrameItems, 44
  - InputProcessor, 44
  - is\_empty, 42
  - is\_full, 42

- Items, 40
- RefFrameItems, 41
- remove, 42
- set, 43
- to\_string, 43
- value, 44
- jeod::RefFrameLinks, 44
  - ~RefFrameLinks, 45
  - default\_path\_size, 46
  - init\_attrjeod\_\_RefFrameLinks, 46
  - InputProcessor, 46
  - operator=, 45
  - RefFrameLinks, 45
- jeod::RefFrameManager, 46
  - ~RefFrameManager, 48
  - add\_frame\_to\_tree, 48
  - add\_ref\_frame, 48
  - check\_ref\_frame\_ownership, 48
  - find\_ref\_frame, 49
  - frame\_is\_subscribed, 49
  - init\_attrjeod\_\_RefFrameManager, 53
  - InputProcessor, 53
  - operator=, 51
  - ref\_frames, 53
  - RefFrameManager, 48
  - remove\_ref\_frame, 51
  - reset\_tree\_root\_node, 51
  - root\_node, 53
  - subscribe\_to\_frame, 51
  - unsubscribe\_to\_frame, 52
  - validate\_name, 52
- jeod::RefFrameMessages, 53
  - attach\_info, 55
  - duplicate\_entry, 55
  - inconsistent\_setup, 55
  - init\_attrjeod\_\_RefFrameMessages, 55
  - InputProcessor, 55
  - internal\_error, 55
  - invalid\_attach, 55
  - invalid\_detach, 56
  - invalid\_enum, 56
  - invalid\_item, 56
  - invalid\_name, 56
  - invalid\_node, 56
  - null\_pointer, 56
  - operator=, 55
  - RefFrameMessages, 55
  - removal\_failed, 56
  - subscription\_error, 57
- jeod::RefFrameOwner, 57
  - ~RefFrameOwner, 57
  - note\_frame\_status\_change, 58
  - RefFrameOwner, 57
- jeod::RefFrameRot, 58
  - ~RefFrameRot, 59
  - ang\_vel\_mag, 61
  - ang\_vel\_this, 61
  - ang\_vel\_unit, 61
  - compute\_ang\_vel\_products, 60
  - compute\_ang\_vel\_unit, 60
  - compute\_quaternion, 60
  - compute\_transformation, 60
  - copy, 60
  - init\_attrjeod\_\_RefFrameRot, 61
  - initialize, 60
  - InputProcessor, 61
  - operator=, 61
  - Q\_parent\_this, 62
  - RefFrameRot, 59
  - T\_parent\_this, 62
- jeod::RefFrameState, 62
  - ~RefFrameState, 64
  - copy, 64
  - decr\_left, 64
  - decr\_right, 64
  - incr\_left, 64
  - incr\_right, 66
  - init\_attrjeod\_\_RefFrameState, 67
  - initialize, 66
  - InputProcessor, 67
  - negate, 66
  - operator=, 66
  - RefFrameState, 63
  - rot, 67
  - trans, 67
- jeod::RefFrameTrans, 67
  - ~RefFrameTrans, 68
  - copy, 69
  - init\_attrjeod\_\_RefFrameTrans, 69
  - initialize, 69
  - InputProcessor, 69
  - operator=, 69
  - position, 69
  - RefFrameTrans, 68
  - velocity, 70
- jeod::SubscribeInterface, 70
  - ~SubscribeInterface, 71
  - unsubscribe, 71
  - subscribe, 71
  - SubscribeInterface, 71
- jeod::Subscription, 71
  - ~Subscription, 73
  - activate, 73
  - active, 76
  - deactivate, 73
  - get\_subscription\_mode, 74
  - init\_attrjeod\_\_Subscription, 75
  - InputProcessor, 75
  - is\_active, 74
  - Mode, 73
  - mode, 76
  - set\_active\_status, 74
  - set\_subscription\_mode, 74
  - subscribe, 75
  - subscribers, 76
  - Subscription, 73

- subscriptions, 75
- unsubscribe, 75
- jeod::TreeLinks
  - ~TreeLinks, 79
  - attach, 79
  - attach\_internal, 79
  - child\_head, 79
  - child\_tail, 80
  - children\_, 87
  - construct\_path\_to\_node, 80
  - container, 80
  - container\_, 87
  - detach, 80
  - detach\_internal, 80
  - find\_last\_common\_index, 81
  - find\_last\_common\_node, 81
  - find\_path\_index, 81
  - has\_children, 81
  - init\_attrjeod\_\_TreeLinks, 86
  - InputProcessor, 86
  - is\_atomic, 82
  - is\_progeny\_of, 82
  - is\_root, 82
  - links\_parent, 82
  - links\_root, 83
  - make\_root, 83
  - nth\_from\_root, 83
  - operator=, 85
  - parent, 85
  - parent\_, 87
  - path\_length, 85
  - path\_to\_node\_, 87
  - reattach, 85
  - root, 86
  - set\_path\_size, 86
  - TreeLinks, 79
  - TreeLinksAscendRange, 86
  - TreeLinksChildrenRange, 86
  - TreeLinksDescentRange, 87
- jeod::TreeLinks< Links, Container, Messages >, 76
- jeod::TreeLinksAscendRange
  - Reverseliterator, 89
  - TreeLinksAscendRange, 89
- jeod::TreeLinksAscendRange< Links >, 88
- jeod::TreeLinksChildIterator< Links, Container >, 89
- jeod::TreeLinksChildrenRange
  - ForwardIterator, 90
  - TreeLinksChildrenRange, 90
- jeod::TreeLinksChildrenRange< Links >, 90
- jeod::TreeLinksDescentIterator< Links, Container >, 91
- jeod::TreeLinksDescentRange
  - ForwardIterator, 91
  - TreeLinksDescentRange, 92
- jeod::TreeLinksDescentRange< Links >, 91
- jeod::TreeLinksIterator< Links, Container >, 92
- jeod::TreeLinksParentIterator< Links, Container >, 92
- jeod::TreeLinksRange
  - begin\_, 93
  - end, 93
  - end\_, 94
  - TreeLinksRange, 93
- jeod::TreeLinksRange< Iterator >, 92
- links
  - jeod::RefFrame, 38
- links\_parent
  - jeod::TreeLinks, 82
- links\_root
  - jeod::TreeLinks, 83
- make\_root
  - jeod::RefFrame, 33
  - jeod::TreeLinks, 83
- Mode
  - jeod::Subscription, 73
- mode
  - jeod::Subscription, 76
- Models, 11
- name
  - jeod::RefFrame, 38
- negate
  - jeod::RefFrameState, 66
- No\_Items
  - jeod::RefFrameItems, 40
- note\_frame\_status\_change
  - jeod::RefFrameOwner, 58
- nth\_from\_root
  - jeod::TreeLinks, 83
- null\_pointer
  - jeod::RefFrameMessages, 56
- operator=
  - jeod::RefFrame, 33
  - jeod::RefFrameLinks, 45
  - jeod::RefFrameManager, 51
  - jeod::RefFrameMessages, 55
  - jeod::RefFrameRot, 61
  - jeod::RefFrameState, 66
  - jeod::RefFrameTrans, 69
  - jeod::TreeLinks, 85
- owner
  - jeod::RefFrame, 38
- parent
  - jeod::TreeLinks, 85
- parent\_
  - jeod::TreeLinks, 87
- path\_length
  - jeod::TreeLinks, 85
- path\_to\_node\_
  - jeod::TreeLinks, 87
- Pos
  - jeod::RefFrameItems, 40
- Pos\_Att
  - jeod::RefFrameItems, 40

- Pos\_Att\_Rate
  - jeod::RefFrameItems, [41](#)
- Pos\_Rate
  - jeod::RefFrameItems, [41](#)
- Pos\_Vel
  - jeod::RefFrameItems, [40](#)
- Pos\_Vel\_Att
  - jeod::RefFrameItems, [40](#)
- Pos\_Vel\_Att\_Rate
  - jeod::RefFrameItems, [41](#)
- Pos\_Vel\_Rate
  - jeod::RefFrameItems, [41](#)
- position
  - jeod::RefFrameTrans, [69](#)
- Q\_parent\_this
  - jeod::RefFrameRot, [62](#)
- Rate
  - jeod::RefFrameItems, [41](#)
- reattach
  - jeod::TreeLinks, [85](#)
- ref\_frame.cc, [96](#)
- ref\_frame.hh, [96](#)
- ref\_frame\_compute\_relative\_state.cc, [97](#)
- ref\_frame\_inline.hh, [97](#)
- ref\_frame\_interface.hh, [98](#)
- ref\_frame\_items.cc, [98](#)
- ref\_frame\_items.hh, [99](#)
- ref\_frame\_items\_inline.hh, [99](#)
- ref\_frame\_links.hh, [99](#)
- ref\_frame\_manager.cc, [100](#)
- ref\_frame\_manager.hh, [100](#)
- ref\_frame\_messages.cc, [101](#)
- ref\_frame\_messages.hh, [102](#)
- ref\_frame\_set\_name.cc, [102](#)
- ref\_frame\_state.cc, [102](#)
- ref\_frame\_state.hh, [103](#)
- ref\_frame\_state\_inline.hh, [103](#)
- ref\_frames
  - jeod::RefFrameManager, [53](#)
- RefFrame
  - jeod::RefFrame, [27](#)
- RefFrameItems
  - jeod::RefFrameItems, [41](#)
- RefFrameLinks
  - jeod::RefFrame, [38](#)
  - jeod::RefFrameLinks, [45](#)
- RefFrameManager
  - jeod::RefFrameManager, [48](#)
- RefFrameMessages
  - jeod::RefFrameMessages, [55](#)
- RefFrameOwner
  - jeod::RefFrameOwner, [57](#)
- RefFrameRot
  - jeod::RefFrameRot, [59](#)
- RefFrameState
  - jeod::RefFrameState, [63](#)
- RefFrameTrans
  - jeod::RefFrameTrans, [68](#)
- RefFrames, [13](#)
- removal\_failed
  - jeod::RefFrameMessages, [56](#)
- remove
  - jeod::RefFrameItems, [42](#)
- remove\_from\_parent
  - jeod::RefFrame, [33](#)
- remove\_ref\_frame
  - jeod::BaseRefFrameManager, [20](#)
  - jeod::RefFrameManager, [51](#)
- reset\_parent
  - jeod::RefFrame, [33](#)
- reset\_tree\_root\_node
  - jeod::BaseRefFrameManager, [22](#)
  - jeod::RefFrameManager, [51](#)
- Reverseliterator
  - jeod::JeodLinksIterators, [23](#)
  - jeod::JeodLinksIterators< const Links >, [24](#)
  - jeod::TreeLinksAscendRange, [89](#)
- root
  - jeod::TreeLinks, [86](#)
- root\_node
  - jeod::RefFrameManager, [53](#)
- rot
  - jeod::RefFrameState, [67](#)
- set
  - jeod::RefFrameItems, [43](#)
- set\_active\_status
  - jeod::RefFrame, [34](#)
  - jeod::Subscription, [74](#)
- set\_name
  - jeod::RefFrame, [34](#), [35](#)
- set\_owner
  - jeod::RefFrame, [37](#)
- set\_path\_size
  - jeod::TreeLinks, [86](#)
- set\_subscription\_mode
  - jeod::Subscription, [74](#)
- set\_timestamp
  - jeod::RefFrame, [37](#)
- state
  - jeod::RefFrame, [38](#)
- Subscribe
  - jeod::Subscription, [73](#)
- subscribe
  - jeod::SubscribeInterface, [71](#)
  - jeod::Subscription, [75](#)
- subscribe\_to\_frame
  - jeod::BaseRefFrameManager, [22](#)
  - jeod::RefFrameManager, [51](#)
- SubscribeInterface
  - jeod::SubscribeInterface, [71](#)
- subscribers
  - jeod::Subscription, [76](#)
- Subscription
  - jeod::Subscription, [73](#)
- subscription.cc, [104](#)

- subscription.hh, [104](#)
- subscription\_error
  - jeod::RefFrameMessages, [57](#)
- subscriptions
  - jeod::Subscription, [75](#)
- T\_parent\_this
  - jeod::RefFrameRot, [62](#)
- timestamp
  - jeod::RefFrame, [37](#)
- to\_string
  - jeod::RefFrameItems, [43](#)
- trans
  - jeod::RefFrameState, [67](#)
- transplant\_node
  - jeod::RefFrame, [37](#)
- tree\_links.hh, [105](#)
- tree\_links\_iterator.hh, [105](#)
- TreeLinks
  - jeod::TreeLinks, [79](#)
- TreeLinksAscendRange
  - jeod::TreeLinks, [86](#)
  - jeod::TreeLinksAscendRange, [89](#)
- TreeLinksChildrenRange
  - jeod::TreeLinks, [86](#)
  - jeod::TreeLinksChildrenRange, [90](#)
- TreeLinksDescentRange
  - jeod::TreeLinks, [87](#)
  - jeod::TreeLinksDescentRange, [92](#)
- TreeLinksRange
  - jeod::TreeLinksRange, [93](#)
- unsubscribe
  - jeod::Subscription, [75](#)
- unsubscribe\_to\_frame
  - jeod::BaseRefFrameManager, [22](#)
  - jeod::RefFrameManager, [52](#)
- update\_time
  - jeod::RefFrame, [39](#)
- Utils, [12](#)
- validate\_name
  - jeod::RefFrameManager, [52](#)
- value
  - jeod::RefFrameItems, [44](#)
- Vel
  - jeod::RefFrameItems, [40](#)
- Vel\_Att
  - jeod::RefFrameItems, [40](#)
- Vel\_Att\_Rate
  - jeod::RefFrameItems, [41](#)
- Vel\_Rate
  - jeod::RefFrameItems, [41](#)
- velocity
  - jeod::RefFrameTrans, [70](#)