

# ContactModel

5.0

Generated by Doxygen 1.8.5

Wed Jun 1 2022 12:05:00



# 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	Interactions . . . . .	12
6.2.1	Detailed Description . . . . .	12
6.3	Contact . . . . .	13
6.3.1	Detailed Description . . . . .	14
6.3.2	Macro Definition Documentation . . . . .	14
6.3.2.1	PATH . . . . .	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::Contact Class Reference . . . . .	17
8.1.1	Detailed Description . . . . .	18
8.1.2	Constructor & Destructor Documentation . . . . .	18
8.1.2.1	Contact . . . . .	18

8.1.2.2	<a href="#">~Contact</a>	18
8.1.2.3	<a href="#">Contact</a>	18
8.1.3	<a href="#">Member Function Documentation</a>	18
8.1.3.1	<a href="#">check_contact</a>	18
8.1.3.2	<a href="#">find_interaction</a>	19
8.1.3.3	<a href="#">initialize_contact</a>	19
8.1.3.4	<a href="#">operator=</a>	19
8.1.3.5	<a href="#">register_contact</a>	19
8.1.3.6	<a href="#">register_contact</a>	19
8.1.3.7	<a href="#">register_contact</a>	20
8.1.3.8	<a href="#">register_contact</a>	21
8.1.3.9	<a href="#">register_interaction</a>	21
8.1.3.10	<a href="#">unique_pair</a>	21
8.1.4	<a href="#">Friends And Related Function Documentation</a>	21
8.1.4.1	<a href="#">init_attrjeod__Contact</a>	21
8.1.4.2	<a href="#">InputProcessor</a>	22
8.1.5	<a href="#">Field Documentation</a>	22
8.1.5.1	<a href="#">active</a>	22
8.1.5.2	<a href="#">contact_limit_factor</a>	22
8.1.5.3	<a href="#">contact_pairs</a>	22
8.1.5.4	<a href="#">dyn_manager</a>	22
8.1.5.5	<a href="#">pair_interactions</a>	22
8.2	<a href="#">jeod::ContactFacet Class Reference</a>	23
8.2.1	<a href="#">Detailed Description</a>	24
8.2.2	<a href="#">Constructor &amp; Destructor Documentation</a>	24
8.2.2.1	<a href="#">ContactFacet</a>	24
8.2.2.2	<a href="#">~ContactFacet</a>	24
8.2.2.3	<a href="#">ContactFacet</a>	24
8.2.3	<a href="#">Member Function Documentation</a>	24
8.2.3.1	<a href="#">calculate_torque</a>	24
8.2.3.2	<a href="#">create_pair</a>	24
8.2.3.3	<a href="#">create_pair</a>	25
8.2.3.4	<a href="#">create_vehicle_point</a>	25
8.2.3.5	<a href="#">get_name</a>	25
8.2.3.6	<a href="#">operator=</a>	25
8.2.3.7	<a href="#">set_max_dimension</a>	25
8.2.4	<a href="#">Friends And Related Function Documentation</a>	25
8.2.4.1	<a href="#">init_attrjeod__ContactFacet</a>	25
8.2.4.2	<a href="#">InputProcessor</a>	25
8.2.5	<a href="#">Field Documentation</a>	25

8.2.5.1	<a href="#">active</a>	25
8.2.5.2	<a href="#">max_dimension</a>	26
8.2.5.3	<a href="#">normal</a>	26
8.2.5.4	<a href="#">position</a>	26
8.2.5.5	<a href="#">surface_type</a>	26
8.2.5.6	<a href="#">vehicle_body</a>	26
8.2.5.7	<a href="#">vehicle_point</a>	27
8.3	<a href="#">jeod::ContactMessages Class Reference</a>	27
8.3.1	<a href="#">Detailed Description</a>	27
8.3.2	<a href="#">Constructor &amp; Destructor Documentation</a>	28
8.3.2.1	<a href="#">ContactMessages</a>	28
8.3.2.2	<a href="#">ContactMessages</a>	28
8.3.3	<a href="#">Member Function Documentation</a>	28
8.3.3.1	<a href="#">operator=</a>	28
8.3.4	<a href="#">Friends And Related Function Documentation</a>	28
8.3.4.1	<a href="#">init_attrjeod__ContactMessages</a>	28
8.3.4.2	<a href="#">InputProcessor</a>	28
8.3.5	<a href="#">Field Documentation</a>	28
8.3.5.1	<a href="#">initialization_error</a>	28
8.3.5.2	<a href="#">initialization_warns</a>	28
8.3.5.3	<a href="#">pre_initialization_error</a>	28
8.3.5.4	<a href="#">runtime_error</a>	29
8.3.5.5	<a href="#">runtime_inform</a>	29
8.3.5.6	<a href="#">runtime_warns</a>	29
8.4	<a href="#">jeod::ContactPair Class Reference</a>	29
8.4.1	<a href="#">Detailed Description</a>	31
8.4.2	<a href="#">Constructor &amp; Destructor Documentation</a>	31
8.4.2.1	<a href="#">ContactPair</a>	31
8.4.2.2	<a href="#">~ContactPair</a>	31
8.4.2.3	<a href="#">ContactPair</a>	31
8.4.3	<a href="#">Member Function Documentation</a>	31
8.4.3.1	<a href="#">check_tree</a>	31
8.4.3.2	<a href="#">get_subject</a>	31
8.4.3.3	<a href="#">get_target</a>	32
8.4.3.4	<a href="#">in_contact</a>	32
8.4.3.5	<a href="#">in_range</a>	32
8.4.3.6	<a href="#">initialize_pair</a>	32
8.4.3.7	<a href="#">initialize_relstate</a>	32
8.4.3.8	<a href="#">is_active</a>	33
8.4.3.9	<a href="#">is_complete</a>	33

8.4.3.10	operator=	33
8.4.4	Friends And Related Function Documentation	33
8.4.4.1	init_attrjeod__ContactPair	33
8.4.4.2	InputProcessor	33
8.4.5	Field Documentation	33
8.4.5.1	interaction	33
8.4.5.2	interaction_distance	33
8.4.5.3	rel_state	34
8.4.5.4	subject	34
8.4.5.5	target	34
8.5	jeod::ContactParams Class Reference	34
8.5.1	Detailed Description	35
8.5.2	Constructor & Destructor Documentation	35
8.5.2.1	ContactParams	35
8.5.2.2	~ContactParams	35
8.5.2.3	ContactParams	35
8.5.3	Member Function Documentation	35
8.5.3.1	operator=	35
8.5.4	Friends And Related Function Documentation	35
8.5.4.1	init_attrjeod__ContactParams	35
8.5.4.2	InputProcessor	35
8.6	jeod::ContactSurface Class Reference	35
8.6.1	Detailed Description	36
8.6.2	Constructor & Destructor Documentation	37
8.6.2.1	ContactSurface	37
8.6.2.2	~ContactSurface	37
8.6.2.3	ContactSurface	37
8.6.3	Member Function Documentation	37
8.6.3.1	allocate_array	37
8.6.3.2	allocate_interaction_facet	37
8.6.3.3	collect_forces_torques	37
8.6.3.4	operator=	37
8.6.4	Friends And Related Function Documentation	38
8.6.4.1	init_attrjeod__ContactSurface	38
8.6.4.2	InputProcessor	38
8.6.5	Field Documentation	38
8.6.5.1	contact_facets	38
8.6.5.2	contact_force	38
8.6.5.3	contact_torque	38
8.6.5.4	facets_size	38

8.7	<a href="#">jeod::ContactSurfaceFactory Class Reference</a>	38
8.7.1	<a href="#">Detailed Description</a>	39
8.7.2	<a href="#">Constructor &amp; Destructor Documentation</a>	39
8.7.2.1	<a href="#">ContactSurfaceFactory</a>	39
8.7.2.2	<a href="#">~ContactSurfaceFactory</a>	40
8.7.2.3	<a href="#">ContactSurfaceFactory</a>	40
8.7.3	<a href="#">Member Function Documentation</a>	40
8.7.3.1	<a href="#">add_facet_params</a>	40
8.7.3.2	<a href="#">create_surface</a>	40
8.7.3.3	<a href="#">operator=</a>	40
8.7.4	<a href="#">Friends And Related Function Documentation</a>	40
8.7.4.1	<a href="#">init_attrjeod__ContactSurfaceFactory</a>	40
8.7.4.2	<a href="#">InputProcessor</a>	40
8.7.5	<a href="#">Field Documentation</a>	40
8.7.5.1	<a href="#">line_contact_facet_factory</a>	40
8.7.5.2	<a href="#">point_contact_facet_factory</a>	41
8.8	<a href="#">jeod::ContactUtils Class Reference</a>	41
8.8.1	<a href="#">Detailed Description</a>	41
8.8.2	<a href="#">Member Function Documentation</a>	41
8.8.2.1	<a href="#">copy_const_char_to_char</a>	41
8.8.2.2	<a href="#">create_relstate_name</a>	42
8.8.2.3	<a href="#">dist_line_segments</a>	42
8.9	<a href="#">jeod::LineContactFacet Class Reference</a>	43
8.9.1	<a href="#">Detailed Description</a>	44
8.9.2	<a href="#">Constructor &amp; Destructor Documentation</a>	44
8.9.2.1	<a href="#">LineContactFacet</a>	44
8.9.2.2	<a href="#">~LineContactFacet</a>	44
8.9.2.3	<a href="#">LineContactFacet</a>	44
8.9.3	<a href="#">Member Function Documentation</a>	44
8.9.3.1	<a href="#">calculate_contact_point</a>	44
8.9.3.2	<a href="#">calculate_torque</a>	44
8.9.3.3	<a href="#">create_pair</a>	45
8.9.3.4	<a href="#">create_pair</a>	45
8.9.3.5	<a href="#">operator=</a>	45
8.9.3.6	<a href="#">set_max_dimension</a>	45
8.9.4	<a href="#">Friends And Related Function Documentation</a>	45
8.9.4.1	<a href="#">init_attrjeod__LineContactFacet</a>	45
8.9.4.2	<a href="#">InputProcessor</a>	45
8.9.5	<a href="#">Field Documentation</a>	46
8.9.5.1	<a href="#">length</a>	46

8.10	<a href="#">jeod::LineContactFacetFactory Class Reference</a>	46
8.10.1	<a href="#">Detailed Description</a>	47
8.10.2	<a href="#">Constructor &amp; Destructor Documentation</a>	47
8.10.2.1	<a href="#">LineContactFacetFactory</a>	47
8.10.2.2	<a href="#">~LineContactFacetFactory</a>	47
8.10.2.3	<a href="#">LineContactFacetFactory</a>	47
8.10.3	<a href="#">Member Function Documentation</a>	47
8.10.3.1	<a href="#">create_facet</a>	47
8.10.3.2	<a href="#">is_correct_factory</a>	47
8.10.3.3	<a href="#">operator=</a>	48
8.10.4	<a href="#">Friends And Related Function Documentation</a>	48
8.10.4.1	<a href="#">init_attrjeod__LineContactFacetFactory</a>	48
8.10.4.2	<a href="#">InputProcessor</a>	48
8.11	<a href="#">jeod::LineContactPair Class Reference</a>	48
8.11.1	<a href="#">Detailed Description</a>	49
8.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	49
8.11.2.1	<a href="#">LineContactPair</a>	49
8.11.2.2	<a href="#">~LineContactPair</a>	49
8.11.2.3	<a href="#">LineContactPair</a>	49
8.11.3	<a href="#">Member Function Documentation</a>	49
8.11.3.1	<a href="#">in_contact</a>	49
8.11.3.2	<a href="#">initialize_pair</a>	49
8.11.3.3	<a href="#">operator=</a>	50
8.11.4	<a href="#">Friends And Related Function Documentation</a>	50
8.11.4.1	<a href="#">init_attrjeod__LineContactPair</a>	50
8.11.4.2	<a href="#">InputProcessor</a>	50
8.11.5	<a href="#">Field Documentation</a>	50
8.11.5.1	<a href="#">line_subject</a>	50
8.11.5.2	<a href="#">line_target</a>	50
8.12	<a href="#">jeod::LinePointContactPair Class Reference</a>	50
8.12.1	<a href="#">Detailed Description</a>	51
8.12.2	<a href="#">Constructor &amp; Destructor Documentation</a>	51
8.12.2.1	<a href="#">LinePointContactPair</a>	51
8.12.2.2	<a href="#">~LinePointContactPair</a>	52
8.12.2.3	<a href="#">LinePointContactPair</a>	52
8.12.3	<a href="#">Member Function Documentation</a>	52
8.12.3.1	<a href="#">in_contact</a>	52
8.12.3.2	<a href="#">initialize_pair</a>	52
8.12.3.3	<a href="#">operator=</a>	52
8.12.4	<a href="#">Friends And Related Function Documentation</a>	52



8.12.4.1	<a href="#">init_attrjeod__LinePointContactPair</a>	52
8.12.4.2	<a href="#">InputProcessor</a>	52
8.12.5	<a href="#">Field Documentation</a>	52
8.12.5.1	<a href="#">line_subject</a>	52
8.12.5.2	<a href="#">point_target</a>	53
8.13	<a href="#">jeod::PairInteraction Class Reference</a>	53
8.13.1	<a href="#">Detailed Description</a>	54
8.13.2	<a href="#">Constructor &amp; Destructor Documentation</a>	54
8.13.2.1	<a href="#">PairInteraction</a>	54
8.13.2.2	<a href="#">~PairInteraction</a>	54
8.13.2.3	<a href="#">PairInteraction</a>	54
8.13.3	<a href="#">Member Function Documentation</a>	54
8.13.3.1	<a href="#">calculate_forces</a>	54
8.13.3.2	<a href="#">is_correct_interaction</a>	54
8.13.3.3	<a href="#">operator=</a>	55
8.13.4	<a href="#">Friends And Related Function Documentation</a>	55
8.13.4.1	<a href="#">init_attrjeod__PairInteraction</a>	55
8.13.4.2	<a href="#">InputProcessor</a>	55
8.13.5	<a href="#">Field Documentation</a>	55
8.13.5.1	<a href="#">friction_mag</a>	55
8.13.5.2	<a href="#">params_1</a>	55
8.13.5.3	<a href="#">params_2</a>	55
8.14	<a href="#">jeod::PointContactFacet Class Reference</a>	56
8.14.1	<a href="#">Detailed Description</a>	57
8.14.2	<a href="#">Constructor &amp; Destructor Documentation</a>	57
8.14.2.1	<a href="#">PointContactFacet</a>	57
8.14.2.2	<a href="#">~PointContactFacet</a>	57
8.14.2.3	<a href="#">PointContactFacet</a>	57
8.14.3	<a href="#">Member Function Documentation</a>	57
8.14.3.1	<a href="#">calculate_contact_point</a>	57
8.14.3.2	<a href="#">calculate_torque</a>	57
8.14.3.3	<a href="#">create_pair</a>	58
8.14.3.4	<a href="#">create_pair</a>	58
8.14.3.5	<a href="#">operator=</a>	58
8.14.3.6	<a href="#">set_max_dimension</a>	58
8.14.4	<a href="#">Friends And Related Function Documentation</a>	59
8.14.4.1	<a href="#">init_attrjeod__PointContactFacet</a>	59
8.14.4.2	<a href="#">InputProcessor</a>	59
8.14.5	<a href="#">Field Documentation</a>	59
8.14.5.1	<a href="#">contact_point</a>	59

8.14.5.2	radius	59
8.15	jeod::PointContactFacetFactory Class Reference	59
8.15.1	Detailed Description	60
8.15.2	Constructor & Destructor Documentation	60
8.15.2.1	PointContactFacetFactory	60
8.15.2.2	~PointContactFacetFactory	60
8.15.2.3	PointContactFacetFactory	61
8.15.3	Member Function Documentation	61
8.15.3.1	create_facet	61
8.15.3.2	is_correct_factory	61
8.15.3.3	operator=	61
8.15.4	Friends And Related Function Documentation	61
8.15.4.1	init_attrjeod__PointContactFacetFactory	61
8.15.4.2	InputProcessor	61
8.16	jeod::PointContactPair Class Reference	62
8.16.1	Detailed Description	62
8.16.2	Constructor & Destructor Documentation	63
8.16.2.1	PointContactPair	63
8.16.2.2	~PointContactPair	63
8.16.2.3	PointContactPair	63
8.16.3	Member Function Documentation	63
8.16.3.1	in_contact	63
8.16.3.2	initialize_pair	63
8.16.3.3	operator=	63
8.16.4	Friends And Related Function Documentation	63
8.16.4.1	init_attrjeod__PointContactPair	63
8.16.4.2	InputProcessor	63
8.16.5	Field Documentation	64
8.16.5.1	point_subject	64
8.16.5.2	point_target	64
8.17	jeod::SpringPairInteraction Class Reference	64
8.17.1	Detailed Description	65
8.17.2	Constructor & Destructor Documentation	65
8.17.2.1	SpringPairInteraction	65
8.17.2.2	~SpringPairInteraction	65
8.17.2.3	SpringPairInteraction	65
8.17.3	Member Function Documentation	65
8.17.3.1	calculate_forces	65
8.17.3.2	operator=	66
8.17.4	Friends And Related Function Documentation	66

8.17.4.1	<a href="#">init_attrjeod__SpringPairInteraction</a>	66
8.17.4.2	<a href="#">InputProcessor</a>	66
8.17.5	<a href="#">Field Documentation</a>	66
8.17.5.1	<a href="#">damping_b</a>	66
8.17.5.2	<a href="#">mu</a>	66
8.17.5.3	<a href="#">spring_k</a>	66
<b>9</b>	<b><a href="#">File Documentation</a></b>	<b>67</b>
9.1	<a href="#">class_declarations.hh File Reference</a>	67
9.1.1	<a href="#">Detailed Description</a>	67
9.2	<a href="#">contact.cc File Reference</a>	67
9.2.1	<a href="#">Detailed Description</a>	67
9.3	<a href="#">contact.hh File Reference</a>	68
9.3.1	<a href="#">Detailed Description</a>	68
9.4	<a href="#">contact_facet.cc File Reference</a>	68
9.4.1	<a href="#">Detailed Description</a>	69
9.5	<a href="#">contact_facet.hh File Reference</a>	69
9.5.1	<a href="#">Detailed Description</a>	69
9.6	<a href="#">contact_messages.cc File Reference</a>	69
9.6.1	<a href="#">Detailed Description</a>	70
9.7	<a href="#">contact_messages.hh File Reference</a>	70
9.7.1	<a href="#">Detailed Description</a>	70
9.8	<a href="#">contact_pair.cc File Reference</a>	70
9.8.1	<a href="#">Detailed Description</a>	70
9.9	<a href="#">contact_pair.hh File Reference</a>	71
9.9.1	<a href="#">Detailed Description</a>	71
9.10	<a href="#">contact_params.cc File Reference</a>	71
9.10.1	<a href="#">Detailed Description</a>	71
9.11	<a href="#">contact_params.hh File Reference</a>	71
9.11.1	<a href="#">Detailed Description</a>	72
9.12	<a href="#">contact_surface.cc File Reference</a>	72
9.12.1	<a href="#">Detailed Description</a>	72
9.13	<a href="#">contact_surface.hh File Reference</a>	72
9.13.1	<a href="#">Detailed Description</a>	73
9.14	<a href="#">contact_surface_factory.cc File Reference</a>	73
9.14.1	<a href="#">Detailed Description</a>	73
9.15	<a href="#">contact_surface_factory.hh File Reference</a>	73
9.15.1	<a href="#">Detailed Description</a>	74
9.16	<a href="#">contact_utils.hh File Reference</a>	74
9.16.1	<a href="#">Detailed Description</a>	74

9.17	<a href="#">contact_utils_inline.hh File Reference</a>	74
9.17.1	<a href="#">Detailed Description</a>	75
9.18	<a href="#">line_contact_facet.cc File Reference</a>	75
9.18.1	<a href="#">Detailed Description</a>	75
9.19	<a href="#">line_contact_facet.hh File Reference</a>	75
9.19.1	<a href="#">Detailed Description</a>	76
9.20	<a href="#">line_contact_facet_factory.cc File Reference</a>	76
9.20.1	<a href="#">Detailed Description</a>	76
9.21	<a href="#">line_contact_facet_factory.hh File Reference</a>	76
9.21.1	<a href="#">Detailed Description</a>	77
9.22	<a href="#">line_contact_pair.cc File Reference</a>	77
9.22.1	<a href="#">Detailed Description</a>	77
9.23	<a href="#">line_contact_pair.hh File Reference</a>	77
9.23.1	<a href="#">Detailed Description</a>	78
9.24	<a href="#">line_point_contact_pair.cc File Reference</a>	78
9.24.1	<a href="#">Detailed Description</a>	78
9.25	<a href="#">line_point_contact_pair.hh File Reference</a>	78
9.25.1	<a href="#">Detailed Description</a>	79
9.26	<a href="#">pair_interaction.cc File Reference</a>	79
9.26.1	<a href="#">Detailed Description</a>	79
9.27	<a href="#">pair_interaction.hh File Reference</a>	79
9.27.1	<a href="#">Detailed Description</a>	79
9.28	<a href="#">point_contact_facet.cc File Reference</a>	80
9.28.1	<a href="#">Detailed Description</a>	80
9.29	<a href="#">point_contact_facet.hh File Reference</a>	80
9.29.1	<a href="#">Detailed Description</a>	81
9.30	<a href="#">point_contact_facet_factory.cc File Reference</a>	81
9.30.1	<a href="#">Detailed Description</a>	81
9.31	<a href="#">point_contact_facet_factory.hh File Reference</a>	81
9.31.1	<a href="#">Detailed Description</a>	82
9.32	<a href="#">point_contact_pair.cc File Reference</a>	82
9.32.1	<a href="#">Detailed Description</a>	82
9.33	<a href="#">point_contact_pair.hh File Reference</a>	82
9.33.1	<a href="#">Detailed Description</a>	83
9.34	<a href="#">spring_pair_interaction.cc File Reference</a>	83
9.34.1	<a href="#">Detailed Description</a>	83
9.35	<a href="#">spring_pair_interaction.hh File Reference</a>	83
9.35.1	<a href="#">Detailed Description</a>	83





# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Models . . . . .	11
Interactions . . . . .	12
Contact . . . . .	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::Contact . . . . .	17
jeod::ContactMessages . . . . .	27
jeod::ContactPair . . . . .	29
jeod::LineContactPair . . . . .	48
jeod::LinePointContactPair . . . . .	50
jeod::PointContactPair . . . . .	62
jeod::ContactUtils . . . . .	41
FacetParams	
jeod::ContactParams . . . . .	34
InteractionFacet	
jeod::ContactFacet . . . . .	23
jeod::PointContactFacet . . . . .	56
jeod::LineContactFacet . . . . .	43
InteractionFacetFactory	
jeod::LineContactFacetFactory . . . . .	46
jeod::PointContactFacetFactory . . . . .	59
InteractionSurface	
jeod::ContactSurface . . . . .	35
InteractionSurfaceFactory	
jeod::ContactSurfaceFactory . . . . .	38
jeod::PairInteraction . . . . .	53
jeod::SpringPairInteraction . . . . .	64



## Chapter 4

# Data Structure Index

### 4.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">jeod::Contact</a>	An base contact class for use in the surface model . . . . .	17
<a href="#">jeod::ContactFacet</a>	An contact interaction specific facet for use in the surface model . . . . .	23
<a href="#">jeod::ContactMessages</a>	Messages associated with use of the contact model . . . . .	27
<a href="#">jeod::ContactPair</a>	An base contact pair class for use in the contact model . . . . .	29
<a href="#">jeod::ContactParams</a>	A base class for all contact parameters used in the surface model . . . . .	34
<a href="#">jeod::ContactSurface</a>	The contact specific interaction surface, for use with the surface model . . . . .	35
<a href="#">jeod::ContactSurfaceFactory</a>	The surface factory that creates an contact specific surface from a general surface . . . . .	38
<a href="#">jeod::ContactUtils</a>	Utility string and math functions for the contact model . . . . .	41
<a href="#">jeod::LineContactFacet</a>	The contact facet based on the distance to a single point, specifically the vehicle point . . . . .	43
<a href="#">jeod::LineContactFacetFactory</a>	Creates a <a href="#">PointContactFacet</a> from an InteractionFacet . . . . .	46
<a href="#">jeod::LineContactPair</a>	An point to point contact pair for use in the contact model . . . . .	48
<a href="#">jeod::LinePointContactPair</a>	An point to point contact pair for use in the contact model . . . . .	50
<a href="#">jeod::PairInteraction</a>	Simple spring contact parameters . . . . .	53
<a href="#">jeod::PointContactFacet</a>	The contact facet based on the distance to a single point, specifically the vehicle point . . . . .	56
<a href="#">jeod::PointContactFacetFactory</a>	Creates a <a href="#">PointContactFacet</a> from an InteractionFacet . . . . .	59
<a href="#">jeod::PointContactPair</a>	An point to point contact pair for use in the contact model . . . . .	62
<a href="#">jeod::SpringPairInteraction</a>	Simple spring contact parameters . . . . .	64



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

<a href="#">class_declarations.hh</a>	Forward declaration of classes defined in the contact model . . . . .	67
<a href="#">contact.cc</a>	Base Contact for use with contact interaction model . . . . .	67
<a href="#">contact.hh</a>	(Base class to for the contact manager for use with contact interaction model) . . . . .	68
<a href="#">contact_facet.cc</a>	Define ContactFacet::create_vehicle_point . . . . .	68
<a href="#">contact_facet.hh</a>	Individual facets for use with contact interaction models . . . . .	69
<a href="#">contact_messages.cc</a>	Implement contact_messages . . . . .	69
<a href="#">contact_messages.hh</a>	Contact message for message handling . . . . .	70
<a href="#">contact_pair.cc</a>	ContactPair class for use with contact interaction model . . . . .	70
<a href="#">contact_pair.hh</a>	Base class for pair of contact facets for use with contact interaction model . . . . .	71
<a href="#">contact_params.cc</a>	Contact parameters for use in the surface model . . . . .	71
<a href="#">contact_params.hh</a>	A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys . . . . .	71
<a href="#">contact_surface.cc</a>	Vehicle surface model for the contact interaction models . . . . .	72
<a href="#">contact_surface.hh</a>	Vehicle surface model for contact . . . . .	72
<a href="#">contact_surface_factory.cc</a>	Factory that creates an contact surface, from a surface model . . . . .	73
<a href="#">contact_surface_factory.hh</a>	Factory that creates an contact interaction surface from a surface model . . . . .	73
<a href="#">contact_utils.hh</a>	This Model is used for utility routines . . . . .	74
<a href="#">contact_utils_inline.hh</a>	Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char . . . . .	74
<a href="#">line_contact_facet.cc</a>	Define LineContactFacet functions . . . . .	75

<a href="#">line_contact_facet.hh</a>	The contact facet based on the distance to a line segment centered on the vehicle point . . . .	75
<a href="#">line_contact_facet_factory.cc</a>	Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object . . . . .	76
<a href="#">line_contact_facet_factory.hh</a>	Creates a line contact facet from an cylinder facet . . . . .	76
<a href="#">line_contact_pair.cc</a>	LineContactPair class for use with contact interaction model . . . . .	77
<a href="#">line_contact_pair.hh</a>	Class for a pair of line contact facets for use with contact interaction model . . . . .	77
<a href="#">line_point_contact_pair.cc</a>	LinePointContactPair class for use with contact interaction model . . . . .	78
<a href="#">line_point_contact_pair.hh</a>	Class for a pair of a line contact facet and a point contact facet for use with contact interaction model . . . . .	78
<a href="#">pair_interaction.cc</a>	A class to define the interaction type for a pair of contact facets . . . . .	79
<a href="#">pair_interaction.hh</a>	A class to define the interaction type for a pair of contact facets . . . . .	79
<a href="#">point_contact_facet.cc</a>	Define PointContactFacet functions . . . . .	80
<a href="#">point_contact_facet.hh</a>	The contact facet based on the distance to a single point, specifically the vehicle point . . . . .	80
<a href="#">point_contact_facet_factory.cc</a>	Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object . . . . .	81
<a href="#">point_contact_facet_factory.hh</a>	Creates a point contact facet from an circular flat plate facet . . . . .	81
<a href="#">point_contact_pair.cc</a>	ContactPair class for use with contact interaction model . . . . .	82
<a href="#">point_contact_pair.hh</a>	Class for a pair of point contact facets for use with contact interaction model . . . . .	82
<a href="#">spring_pair_interaction.cc</a>	Spring pair interaction for use in the contact model . . . . .	83
<a href="#">spring_pair_interaction.hh</a>	A class for pair interactions based on a simple spring . . . . .	83



## Chapter 6

# Module Documentation

### 6.1 Models

#### Modules

- [Interactions](#)

#### 6.1.1 Detailed Description

## 6.2 Interactions

### Modules

- [Contact](#)

### 6.2.1 Detailed Description

## 6.3 Contact

### Files

- file [class\\_declarations.hh](#)  
*Forward declaration of classes defined in the contact model.*
- file [contact.hh](#)  
*(Base class to for the contact manager for use with contact interaction model)*
- file [contact\\_facet.hh](#)  
*Individual facets for use with contact interaction models.*
- file [contact\\_messages.hh](#)  
*Contact message for message handling.*
- file [contact\\_pair.hh](#)  
*Base class for pair of contact facets for use with contact interaction model.*
- file [contact\\_params.hh](#)  
*A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.*
- file [contact\\_surface.hh](#)  
*Vehicle surface model for contact.*
- file [contact\\_surface\\_factory.hh](#)  
*Factory that creates an contact interaction surface from a surface model.*
- file [contact\\_utils.hh](#)  
*This Model is used for utility routines.*
- file [contact\\_utils\\_inline.hh](#)  
*Define ContactUtils::create\_relstate\_name, ContactUtils::copy\_const\_char\_to\_char.*
- file [line\\_contact\\_facet.hh](#)  
*The contact facet based on the distance to a line segment centered on the vehicle point.*
- file [line\\_contact\\_facet\\_factory.hh](#)  
*Creates a line contact facet from an cylinder facet.*
- file [line\\_contact\\_pair.hh](#)  
*Class for a pair of line contact facets for use with contact interaction model.*
- file [line\\_point\\_contact\\_pair.hh](#)  
*Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.*
- file [pair\\_interaction.hh](#)  
*A class to define the interaction type for a pair of contact facets.*
- file [point\\_contact\\_facet.hh](#)  
*The contact facet based on the distance to a single point, specifically the vehicle point.*
- file [point\\_contact\\_facet\\_factory.hh](#)  
*Creates a point contact facet from an circular flat plate facet.*
- file [point\\_contact\\_pair.hh](#)  
*Class for a pair of point contact facets for use with contact interaction model.*
- file [spring\\_pair\\_interaction.hh](#)  
*A class for pair interactions based on a simple spring.*
- file [contact.cc](#)  
*Base Contact for use with contact interaction model.*
- file [contact\\_facet.cc](#)  
*Define ContactFacet::create\_vehicle\_point.*
- file [contact\\_messages.cc](#)  
*Implement contact\_messages.*
- file [contact\\_pair.cc](#)  
*ContactPair class for use with contact interaction model.*
- file [contact\\_params.cc](#)

- contact parameters for use in the surface model*
- file [contact\\_surface.cc](#)
  - Vehicle surface model for the contact interaction models.*
- file [contact\\_surface\\_factory.cc](#)
  - Factory that creates an contact surface, from a surface model.*
- file [line\\_contact\\_facet.cc](#)
  - Define LineContactFacet functions.*
- file [line\\_contact\\_facet\\_factory.cc](#)
  - Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.*
- file [line\\_contact\\_pair.cc](#)
  - LineContactPair class for use with contact interaction model.*
- file [line\\_point\\_contact\\_pair.cc](#)
  - LinePointContactPair class for use with contact interaction model.*
- file [pair\\_interaction.cc](#)
  - A class to define the interaction type for a pair of contact facets.*
- file [point\\_contact\\_facet.cc](#)
  - Define PointContactFacet functions.*
- file [point\\_contact\\_facet\\_factory.cc](#)
  - Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.*
- file [point\\_contact\\_pair.cc](#)
  - ContactPair class for use with contact interaction model.*
- file [spring\\_pair\\_interaction.cc](#)
  - spring pair interaction for use in the contact model*

## Namespaces

- [jeod](#)
  - Namespace jeod.*

## Macros

- `#define` [PATH](#) "interactions/contact"

### 6.3.1 Detailed Description

### 6.3.2 Macro Definition Documentation

#### 6.3.2.1 `#define` [PATH](#) "interactions/contact"

Definition at line 37 of file `contact_messages.cc`.

## Chapter 7

# Namespace Documentation

### 7.1 jeod Namespace Reference

Namespace jeod.

#### Data Structures

- class [Contact](#)  
*An base contact class for use in the surface model.*
- class [ContactFacet](#)  
*An contact interaction specific facet for use in the surface model.*
- class [ContactMessages](#)  
*Messages associated with use of the contact model.*
- class [ContactPair](#)  
*An base contact pair class for use in the contact model.*
- class [ContactParams](#)  
*A base class for all contact parameters used in the surface model.*
- class [ContactSurface](#)  
*The contact specific interaction surface, for use with the surface model.*
- class [ContactSurfaceFactory](#)  
*The surface factory that creates an contact specific surface from a general surface.*
- class [ContactUtils](#)  
*Utility string and math functions for the contact model.*
- class [LineContactFacet](#)  
*The contact facet based on the distance to a single point, specifically the vehicle point.*
- class [LineContactFacetFactory](#)  
*Creates a [PointContactFacet](#) from an [InteractionFacet](#).*
- class [LineContactPair](#)  
*An point to point contact pair for use in the contact model.*
- class [LinePointContactPair](#)  
*An point to point contact pair for use in the contact model.*
- class [PairInteraction](#)  
*Simple spring contact parameters.*
- class [PointContactFacet](#)  
*The contact facet based on the distance to a single point, specifically the vehicle point.*
- class [PointContactFacetFactory](#)  
*Creates a [PointContactFacet](#) from an [InteractionFacet](#).*

- class [PointContactPair](#)  
*An point to point contact pair for use in the contact model.*
- class [SpringPairInteraction](#)  
*Simple spring contact parameters.*

### 7.1.1 Detailed Description

Namespace jeod.

## Chapter 8

# Data Structure Documentation

### 8.1 jeod::Contact Class Reference

An base contact class for use in the surface model.

```
#include <contact.hh>
```

#### Public Member Functions

- [Contact](#) ()  
*Default Constructor.*
- virtual [~Contact](#) ()  
*Destructor.*
- void [register\\_contact](#) ([ContactFacet](#) \*facet)  
*Register one [ContactFacet](#) with all inclusive interactions with other registered [ContactFacets](#).*
- void [register\\_contact](#) ([ContactFacet](#) \*\*facets, unsigned int n\_facets)  
*Register an array of [ContactFacets](#) with all inclusive interactions with other registered [ContactFacets](#).*
- void [register\\_contact](#) ([ContactFacet](#) \*facet1, [ContactFacet](#) \*facet2)  
*Register two facets as a pair.*
- void [register\\_contact](#) ([ContactFacet](#) \*\*facets1, unsigned int n\_facets1, [ContactFacet](#) \*\*facets2, unsigned int n\_facets2)  
*Register to arrays of facets and create specific pairs between all of them.*
- void [register\\_interaction](#) ([PairInteraction](#) \*interaction)  
*Register a pair interaction.*
- virtual [PairInteraction](#) \* [find\\_interaction](#) ([ContactParams](#) \*params\_1, [ContactParams](#) \*params\_2)  
*find a [PairInteraction](#) based on a set of [ContactParams](#).*
- void [initialize\\_contact](#) ([DynManager](#) \*manager)  
*Initialize [ContactFacets](#) and the manager by cleaning up the pair list.*
- bool [unique\\_pair](#) ([ContactFacet](#) \*facet\_1, [ContactFacet](#) \*facet\_2)  
*Check to see if a pair of facets already exists.*
- void [check\\_contact](#) ()  
*iterate through contact pairs list then call the appropriate contact resolution functions*

#### Data Fields

- bool [active](#)  
*toggles contact on and off, true=on false=off*
- double [contact\\_limit\\_factor](#)  
*factor determines if contact is limited by a multiple of the maximum dimensions of the facets in a pair.*

## Protected Attributes

- DynManager \* [dyn\\_manager](#)  
*Pointer to the dyn\_manager so relstates and be successfully initialized.*
- JeodPointerList< [ContactPair](#) >  
::type [contact\\_pairs](#)  
*list of all possible pairings of contact facets registered with this contact class or derived class*
- JeodPointerList  
< [PairInteraction](#) >::type [pair\\_interactions](#)  
*list of all possible pair interaction types*

## Private Member Functions

- [Contact](#) & [operator=](#) (const [Contact](#) &rhs)
- [Contact](#) (const [Contact](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_Contact](#) ()

### 8.1.1 Detailed Description

An base contact class for use in the surface model.

Definition at line 56 of file [contact.hh](#).

### 8.1.2 Constructor & Destructor Documentation

#### 8.1.2.1 `jeod::Contact::Contact ( void )`

Default Constructor.

Definition at line 52 of file [contact.cc](#).

References [contact\\_pairs](#), and [pair\\_interactions](#).

#### 8.1.2.2 `jeod::Contact::~~Contact ( void )` `[virtual]`

Destructor.

Definition at line 69 of file [contact.cc](#).

References [contact\\_pairs](#), and [pair\\_interactions](#).

#### 8.1.2.3 `jeod::Contact::Contact ( const Contact & rhs )` `[private]`

### 8.1.3 Member Function Documentation

#### 8.1.3.1 `void jeod::Contact::check_contact ( void )`

iterate through contact pairs list then call the appropriate contact resolution functions

Definition at line 97 of file [contact.cc](#).

References [active](#), and [contact\\_pairs](#).



**8.1.3.2** `PairInteraction * jeod::Contact::find_interaction ( ContactParams * params_1, ContactParams * params_2 )`  
`[virtual]`

find a [PairInteraction](#) based on a set of [ContactParams](#).

Returns

pointer to a [PairInteraction](#)

Parameters

in	<i>params_1</i>	<a href="#">ContactParams</a> from a <a href="#">ContactFacet</a>
in	<i>params_2</i>	<a href="#">ContactParams</a> from a <a href="#">ContactFacet</a>

Definition at line 281 of file `contact.cc`.

References `pair_interactions`.

Referenced by `jeod::LineContactFacet::create_pair()`, and `jeod::PointContactFacet::create_pair()`.

**8.1.3.3** `void jeod::Contact::initialize_contact ( DynManager * manager )`

Initialize `ContactFacets` and the manager by cleaning up the pair list.

Parameters

in, out	<i>manager</i>	Dynamics Manager
---------	----------------	------------------

Definition at line 119 of file `contact.cc`.

References `contact_pairs`, `jeod::ContactFacet::create_pair()`, `dyn_manager`, `jeod::ContactPair::get_subject()`, `jeod::ContactPair::initialize_relstate()`, and `unique_pair()`.

**8.1.3.4** `Contact& jeod::Contact::operator= ( const Contact & rhs )` `[private]`

**8.1.3.5** `void jeod::Contact::register_contact ( ContactFacet * facet )`

Register one [ContactFacet](#) with all inclusive interactions with other registered `ContactFacets`.

Parameters

in, out	<i>facet</i>	<a href="#">ContactFacet</a>
---------	--------------	------------------------------

Definition at line 178 of file `contact.cc`.

References `contact_pairs`, and `jeod::ContactFacet::create_pair()`.

Referenced by `register_contact()`.

**8.1.3.6** `void jeod::Contact::register_contact ( ContactFacet ** facets, unsigned int nFacets )`

Register an array of `ContactFacets` with all inclusive interactions with other registered `ContactFacets`.

Parameters

in, out	<i>facets</i>	array of <code>ContactFacets</code>
in	<i>nFacets</i>	number of <code>ContactFacets</code> in array

Definition at line 199 of file `contact.cc`.

References `register_contact()`.

8.1.3.7 void jeod::Contact::register\_contact ( ContactFacet \* *facet1*, ContactFacet \* *facet2* )

Register two facets as a pair.

## Parameters

in, out	<i>facet1</i>	<a href="#">Contact</a> Facet 1
in, out	<i>facet2</i>	<a href="#">Contact</a> Facet 2

Definition at line 217 of file contact.cc.

References `contact_pairs`, `jeod::ContactFacet::create_pair()`, and `dyn_manager`.

**8.1.3.8** `void jeod::Contact::register_contact ( ContactFacet ** facets1, unsigned int nFacets1, ContactFacet ** facets2, unsigned int nFacets2 )`

Register to arrays of facets and create specific pairs between all of them.

## Parameters

in, out	<i>facets1</i>	array of <a href="#">ContactFacets</a>
in	<i>nFacets1</i>	number of <a href="#">ContactFacets</a> in array
in, out	<i>facets2</i>	array of <a href="#">ContactFacets</a>
in	<i>nFacets2</i>	number of <a href="#">ContactFacets</a> in array

Definition at line 244 of file contact.cc.

References `register_contact()`.

**8.1.3.9** `void jeod::Contact::register_interaction ( PairInteraction * interaction )`

Register a pair interaction.

## Parameters

in	<i>interaction</i>	<a href="#">PairInteraction</a> to add to list
----	--------------------	--

Definition at line 266 of file contact.cc.

References `pair_interactions`.

**8.1.3.10** `bool jeod::Contact::unique_pair ( ContactFacet * facet_1, ContactFacet * facet_2 )`

Check to see if a pair of facets already exists.

## Returns

bool

## Parameters

in, out	<i>facet_1</i>	<a href="#">ContactFacet</a>
in, out	<i>facet_2</i>	<a href="#">ContactFacet</a>

Definition at line 155 of file contact.cc.

References `contact_pairs`.

Referenced by `initialize_contact()`.

## 8.1.4 Friends And Related Function Documentation

**8.1.4.1** `void init_attrjeod__Contact ( ) [friend]`

#### 8.1.4.2 friend class InputProcessor [friend]

Definition at line 58 of file contact.hh.

### 8.1.5 Field Documentation

#### 8.1.5.1 bool jeod::Contact::active

toggles contact on and off, true=on false=off

trick\_units(-)

Definition at line 64 of file contact.hh.

Referenced by check\_contact().

#### 8.1.5.2 double jeod::Contact::contact\_limit\_factor

factor determines if contact is limited by a multiple of the maximum dimensions of the facets in a pair.

trick\_units(-)

Definition at line 70 of file contact.hh.

Referenced by jeod::LineContactFacet::create\_pair(), and jeod::PointContactFacet::create\_pair().

#### 8.1.5.3 JeodPointerList<ContactPair>::type jeod::Contact::contact\_pairs [protected]

list of all possible pairings of contact facets registered with this contact class or derived class

trick\_io(\*\*)

Definition at line 129 of file contact.hh.

Referenced by check\_contact(), Contact(), initialize\_contact(), register\_contact(), unique\_pair(), and ~Contact().

#### 8.1.5.4 DynManager\* jeod::Contact::dyn\_manager [protected]

Pointer to the dyn\_manager so relstates and be successfully initialized.

trick\_units(-)

Definition at line 123 of file contact.hh.

Referenced by initialize\_contact(), and register\_contact().

#### 8.1.5.5 JeodPointerList<PairInteraction>::type jeod::Contact::pair\_interactions [protected]

list of all possible pair interaction types

trick\_io(\*\*)

Definition at line 134 of file contact.hh.

Referenced by Contact(), find\_interaction(), register\_interaction(), and ~Contact().

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

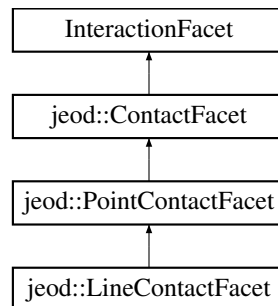
- [contact.hh](#)
- [contact.cc](#)

## 8.2 jeod::ContactFacet Class Reference

An contact interaction specific facet for use in the surface model.

```
#include <contact_facet.hh>
```

Inheritance diagram for jeod::ContactFacet:



### Public Member Functions

- [ContactFacet](#) ()  
*Default constructor.*
- virtual [~ContactFacet](#) ()  
*Destructor.*
- void [create\\_vehicle\\_point](#) (void)  
*Create a vehicle point to track the state information of the contact facet.*
- virtual void [set\\_max\\_dimension](#) (void)=0  
*Calculate the max dimension of the facet for range limit determination.*
- const char \* [get\\_name](#) (void) const  
*Accessor for name.*
- virtual void [calculate\\_torque](#) (double \*tmp\_force)=0  
*Calculate the torque acting on the facet in the vehicle structural frame.*
- virtual [ContactPair](#) \* [create\\_pair](#) (void)=0  
*Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*
- virtual [ContactPair](#) \* [create\\_pair](#) ([ContactFacet](#) \*target, [Contact](#) \*contact)=0  
*Overloaded functions that creates a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*

### Data Fields

- bool [active](#)  
*toggles this contact facet on and off, true=on false=off*
- [ContactParams](#) \* [surface\\_type](#)  
*Stores the name of surface material that the facet is constructed of.*
- DynBody \* [vehicle\\_body](#)  
*DynBody associated with this facet for structural frame information.*
- double [max\\_dimension](#)  
*maximum dimension of the contact facet for use in limiting pair in\_contact calls*
- double [position](#) [3]  
*position of the facet in vehicle structural frame.*
- double [normal](#) [3]

*normal of the facet relative to the vehicle structural frame.*

- `const BodyRefFrame * vehicle_point`

*Vehicle point for relstate calculations.*

## Private Member Functions

- `ContactFacet & operator=` (const `ContactFacet` &rhs)
- `ContactFacet` (const `ContactFacet` &rhs)

## Friends

- class `InputProcessor`
- void `init_attrjeod__ContactFacet` ()

### 8.2.1 Detailed Description

An contact interaction specific facet for use in the surface model.

Definition at line 50 of file `contact_facet.hh`.

### 8.2.2 Constructor & Destructor Documentation

#### 8.2.2.1 `jeod::ContactFacet::ContactFacet ( void )`

Default constructor.

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

References `normal`, and `position`.

#### 8.2.2.2 `jeod::ContactFacet::~~ContactFacet ( void )` `[virtual]`

Destructor.

Definition at line 65 of file `contact_facet.cc`.

#### 8.2.2.3 `jeod::ContactFacet::ContactFacet ( const ContactFacet & rhs )` `[private]`

### 8.2.3 Member Function Documentation

#### 8.2.3.1 `virtual void jeod::ContactFacet::calculate_torque ( double * tmp_force )` `[pure virtual]`

Calculate the torque acting on the facet in the vehicle structural frame.

Implemented in `jeod::PointContactFacet`, and `jeod::LineContactFacet`.

Referenced by `jeod::SpringPairInteraction::calculate_forces()`.

#### 8.2.3.2 `virtual ContactPair* jeod::ContactFacet::create_pair ( void )` `[pure virtual]`

Overloaded functions that create a `ContactPair` and pass the address of it to the `Contact` class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

Implemented in `jeod::PointContactFacet`, and `jeod::LineContactFacet`.

Referenced by `jeod::Contact::initialize_contact()`, and `jeod::Contact::register_contact()`.

**8.2.3.3** `virtual ContactPair* jeod::ContactFacet::create_pair ( ContactFacet * target, Contact * contact )` `[pure virtual]`

Overloaded functions that creates a `ContactPair` and pass the address of it to the `Contact` class for addition to the list of pairs.

This function is called when a subject and target are known.

Implemented in `jeod::PointContactFacet`, and `jeod::LineContactFacet`.

**8.2.3.4** `void jeod::ContactFacet::create_vehicle_point ( void )`

Create a vehicle point to track the state information of the contact facet.

Create a vehicle point from the base facet position and orientation and store the created vehicle point in the contact facet.

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

References `get_name()`, `normal`, `position`, `vehicle_body`, and `vehicle_point`.

Referenced by `jeod::LineContactFacetFactory::create_facet()`, and `jeod::PointContactFacetFactory::create_facet()`.

**8.2.3.5** `const char * jeod::ContactFacet::get_name ( void ) const` `[inline]`

Accessor for name.

Returns

Point name

Definition at line 153 of file `contact_facet.hh`.

Referenced by `create_vehicle_point()`.

**8.2.3.6** `ContactFacet& jeod::ContactFacet::operator= ( const ContactFacet & rhs )` `[private]`

**8.2.3.7** `virtual void jeod::ContactFacet::set_max_dimension ( void )` `[pure virtual]`

Calculate the max dimension of the facet for range limit determination.

Implemented in `jeod::PointContactFacet`, and `jeod::LineContactFacet`.

## 8.2.4 Friends And Related Function Documentation

**8.2.4.1** `void init_attrjeod__ContactFacet ( )` `[friend]`

**8.2.4.2** `friend class InputProcessor` `[friend]`

Definition at line 52 of file `contact_facet.hh`.

## 8.2.5 Field Documentation

**8.2.5.1** `bool jeod::ContactFacet::active`

toggles this contact facet on and off, true=on false=off

trick\_units(-)

Definition at line 58 of file contact\_facet.hh.

Referenced by jeod::ContactPair::is\_active().

#### 8.2.5.2 double jeod::ContactFacet::max\_dimension

maximum dimension of the contact facet for use in limiting pair in \_contact calls

trick\_units(m)

Definition at line 75 of file contact\_facet.hh.

Referenced by jeod::LineContactFacet::create\_pair(), jeod::PointContactFacet::create\_pair(), jeod::LineContactFacet::set\_max\_dimension(), and jeod::PointContactFacet::set\_max\_dimension().

#### 8.2.5.3 double jeod::ContactFacet::normal[3]

normal of the facet relative to the vehicle structural frame.

trick\_units(-)

Definition at line 85 of file contact\_facet.hh.

Referenced by ContactFacet(), jeod::PointContactFacetFactory::create\_facet(), jeod::LineContactFacetFactory::create\_facet(), and create\_vehicle\_point().

#### 8.2.5.4 double jeod::ContactFacet::position[3]

position of the facet in vehicle structural frame.

trick\_units(m)

Definition at line 80 of file contact\_facet.hh.

Referenced by ContactFacet(), jeod::PointContactFacetFactory::create\_facet(), jeod::LineContactFacetFactory::create\_facet(), and create\_vehicle\_point().

#### 8.2.5.5 ContactParams\* jeod::ContactFacet::surface\_type

Stores the name of surface material that the facet is constructed of.

This information is used to determine the contact parameters used when pairs are constructed. trick\_units(-)

Definition at line 65 of file contact\_facet.hh.

Referenced by jeod::LineContactFacetFactory::create\_facet(), jeod::PointContactFacetFactory::create\_facet(), jeod::LineContactFacet::create\_pair(), and jeod::PointContactFacet::create\_pair().

#### 8.2.5.6 DynBody\* jeod::ContactFacet::vehicle\_body

DynBody associated with this facet for structural frame information.

trick\_units(-)

Definition at line 70 of file contact\_facet.hh.

Referenced by jeod::SpringPairInteraction::calculate\_forces(), jeod::LineContactFacet::calculate\_torque(), jeod::PointContactFacet::calculate\_torque(), jeod::LineContactFacetFactory::create\_facet(), jeod::PointContactFacetFactory::create\_facet(), create\_vehicle\_point(), and jeod::ContactPair::initialize\_relstate().



#### 8.2.5.7 const BodyRefFrame\* jeod::ContactFacet::vehicle\_point

Vehicle point for relstate calculations.

trick\_units(-)

Definition at line 90 of file contact\_facet.hh.

Referenced by jeod::SpringPairInteraction::calculate\_forces(), jeod::LineContactFacet::calculate\_torque(), jeod::PointContactFacet::calculate\_torque(), create\_vehicle\_point(), jeod::PointContactPair::initialize\_pair(), jeod::LineContactPair::initialize\_pair(), and jeod::LinePointContactPair::initialize\_pair().

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

- [contact\\_facet.hh](#)
- [contact\\_facet.cc](#)

## 8.3 jeod::ContactMessages Class Reference

Messages associated with use of the contact model.

```
#include <contact_messages.hh>
```

### Static Public Attributes

- static char const \* [initialization\\_error](#)  
*Associated with errors during initialization of the contact model.*
- static char const \* [runtime\\_error](#)  
*Associated with errors during the runtime of the contact model.*
- static char const \* [pre\\_initialization\\_error](#)  
*Associated with errors during the setup of the system, before runtime.*
- static char const \* [initialization\\_warns](#)  
*Associated with warning during initialization of the contact model.*
- static char const \* [runtime\\_warns](#)  
*Associated with warnings given at runtime.*
- static char const \* [runtime\\_inform](#)  
*Associated with information given at runtime.*

### Private Member Functions

- [ContactMessages](#) (void)
- [ContactMessages](#) (const [ContactMessages](#) &rhs)
- [ContactMessages](#) & operator= (const [ContactMessages](#) &rhs)

### Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_ContactMessages](#) ()

#### 8.3.1 Detailed Description

Messages associated with use of the contact model.

Definition at line 50 of file contact\_messages.hh.

### 8.3.2 Constructor & Destructor Documentation

8.3.2.1 `jeod::ContactMessages::ContactMessages ( void ) [private]`

8.3.2.2 `jeod::ContactMessages::ContactMessages ( const ContactMessages & rhs ) [private]`

### 8.3.3 Member Function Documentation

8.3.3.1 `ContactMessages& jeod::ContactMessages::operator= ( const ContactMessages & rhs ) [private]`

### 8.3.4 Friends And Related Function Documentation

8.3.4.1 `void init_attrjeod__ContactMessages ( ) [friend]`

8.3.4.2 `friend class InputProcessor [friend]`

Definition at line 53 of file `contact_messages.hh`.

### 8.3.5 Field Documentation

8.3.5.1 `char const * jeod::ContactMessages::initialization_error [static]`

#### Initial value:

```
=
    "interactions/contact" "initialization_error"
```

Associated with errors during initialization of the contact model.

`trick_units(-)`

Definition at line 64 of file `contact_messages.hh`.

Referenced by `jeod::ContactSurface::allocate_array()`, `jeod::ContactSurface::allocate_interaction_facet()`, `jeod::LineContactFacetFactory::create_facet()`, `jeod::PointContactFacetFactory::create_facet()`, and `jeod::ContactSurfaceFactory::create_surface()`.

8.3.5.2 `char const * jeod::ContactMessages::initialization_warns [static]`

#### Initial value:

```
=
    "interactions/contact" "initialization_warns"
```

Associated with warning during initialization of the contact model.

`trick_units(-)`

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

Referenced by `jeod::LineContactFacet::create_pair()`, and `jeod::PointContactFacet::create_pair()`.

8.3.5.3 `char const * jeod::ContactMessages::pre_initialization_error [static]`

#### Initial value:

```
=
    "interactions/contact" "pre_initialization_error"
```

Associated with errors during the setup of the system, before runtime.

trick\_units(–)

Definition at line 72 of file contact\_messages.hh.

Referenced by jeod::ContactSurfaceFactory::add\_facet\_params().

#### 8.3.5.4 char const \* jeod::ContactMessages::runtime\_error [static]

**Initial value:**

```
=
    "interactions/contact" "runtime_error"
```

Associated with errors during the runtime of the contact model.

trick\_units(–)

Definition at line 68 of file contact\_messages.hh.

#### 8.3.5.5 char const \* jeod::ContactMessages::runtime\_inform [static]

**Initial value:**

```
=
    "interactions/contact" "runtime_warns"
```

Associated with information given at runtime.

trick\_units(–)

Definition at line 90 of file contact\_messages.hh.

#### 8.3.5.6 char const \* jeod::ContactMessages::runtime\_warns [static]

**Initial value:**

```
=
    "interactions/contact" "runtime_warns"
```

Associated with warnings given at runtime.

trick\_units(–)

Definition at line 83 of file contact\_messages.hh.

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

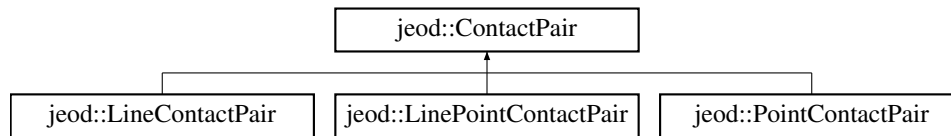
- [contact\\_messages.hh](#)
- [contact\\_messages.cc](#)

## 8.4 jeod::ContactPair Class Reference

An base contact pair class for use in the contact model.

```
#include <contact_pair.hh>
```

Inheritance diagram for jeod::ContactPair:



## Public Member Functions

- [ContactPair](#) ()  
*Default Constructor.*
- virtual [~ContactPair](#) ()  
*Destructor.*
- bool [in\\_range](#) ()  
*test whether the pair is in range for interaction*
- bool [is\\_active](#) ()  
*Determine if contact can occur between the two facets.*
- bool [is\\_complete](#) (void)  
*Determine if the pair has a target facet.*
- [ContactFacet](#) \* [get\\_subject](#) (void)  
*Determine if the pair has a target facet.*
- [ContactFacet](#) \* [get\\_target](#) (void)  
*Determine if the pair has a target facet.*
- virtual void [in\\_contact](#) (void)=0  
*Virtual funtion to determine if the pair is in contact.*
- virtual void [initialize\\_pair](#) ([ContactFacet](#) \*subject\_facet, [ContactFacet](#) \*target\_facet)=0  
*Initialize the contact pair by setting the subject, target, and creating the relstate if possible.*
- virtual void [initialize\\_relstate](#) (DynManager \*dyn\_manager)  
*Initialize the relative state between the facets and register with the dynamics manager.*
- virtual bool [check\\_tree](#) ()  
*Make sure the two contact facets are not on the same mass tree.*

## Data Fields

- [PairInteraction](#) \* [interaction](#)  
*Parameters that define the force calculation function between the subjec and target.*
- double [interaction\\_distance](#)  
*rel\_state distance at which in\_contact should be called*

## Protected Attributes

- RelativeDerivedState [rel\\_state](#)  
*Current relative state between the subject and the target in the subject frame.*
- [ContactFacet](#) \* [subject](#)  
*pointer to the contact facet that is the subject of the associated relative states.*
- [ContactFacet](#) \* [target](#)  
*pointer to the contact facet that is the target of the associated relative states.*

## Private Member Functions

- [ContactPair](#) & [operator=](#) (const [ContactPair](#) &rhs)
- [ContactPair](#) (const [ContactPair](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_ContactPair](#) ()

### 8.4.1 Detailed Description

An base contact pair class for use in the contact model.

Definition at line 49 of file `contact_pair.hh`.

### 8.4.2 Constructor & Destructor Documentation

#### 8.4.2.1 `jeod::ContactPair::ContactPair ( void )`

Default Constructor.

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

#### 8.4.2.2 `jeod::ContactPair::~~ContactPair ( void )` [virtual]

Destructor.

Definition at line 60 of file `contact_pair.cc`.

#### 8.4.2.3 `jeod::ContactPair::ContactPair ( const ContactPair & rhs )` [private]

### 8.4.3 Member Function Documentation

#### 8.4.3.1 `bool jeod::ContactPair::check_tree ( void )` [virtual]

Make sure the two contact facets are not on the same mass tree.

##### Returns

`bool`

Definition at line 136 of file `contact_pair.cc`.

References `subject`, and `target`.

Referenced by `is_active()`.

#### 8.4.3.2 `ContactFacet * jeod::ContactPair::get_subject ( void )`

Determine if the pair has a target facet.

##### Returns

subject [ContactFacet](#)

Definition at line 114 of file `contact_pair.cc`.

References `subject`.

Referenced by `jeod::Contact::initialize_contact()`.

#### 8.4.3.3 **ContactFacet \* jeod::ContactPair::get\_target ( void )**

Determine if the pair has a target facet.

##### Returns

target [ContactFacet](#)

Definition at line 125 of file `contact_pair.cc`.

References `target`.

#### 8.4.3.4 **virtual void jeod::ContactPair::in\_contact ( void )** [pure virtual]

Virtual funtion to determine if the pair is in contact.

[Contact](#) depends on specific geometry so implementation has to wait for a derived class.

Implemented in [jeod::LinePointContactPair](#), [jeod::LineContactPair](#), and [jeod::PointContactPair](#).

#### 8.4.3.5 **bool jeod::ContactPair::in\_range ( void )**

test whether the pair is in range for interaction

##### Returns

bool

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

References `interaction_distance`, and `rel_state`.

#### 8.4.3.6 **virtual void jeod::ContactPair::initialize\_pair ( ContactFacet \* subject\_facet, ContactFacet \* target\_facet )** [pure virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

##### Parameters

in, out	<i>subject_facet</i>	subject <a href="#">ContactFacet</a>
in, out	<i>target_facet</i>	target <a href="#">ContactFacet</a>

Implemented in [jeod::LinePointContactPair](#), [jeod::LineContactPair](#), and [jeod::PointContactPair](#).

#### 8.4.3.7 **void jeod::ContactPair::initialize\_relstate ( DynManager \* dyn\_manager )** [virtual]

Initialize the relative state between the facets and register with the dynamics manager.

Initialize the relstate using the DynManager provided by [Contact](#) class.

##### Parameters

in	<i>dyn_manager</i>	dynamics manager
----	--------------------	------------------

Definition at line 158 of file `contact_pair.cc`.

References `rel_state`, `subject`, and `jeod::ContactFacet::vehicle_body`.

Referenced by `jeod::Contact::initialize_contact()`.

**8.4.3.8** `bool jeod::ContactPair::is_active ( void )`

Determine if contact can occur between the two facets.

Returns

`bool`

Definition at line 86 of file `contact_pair.cc`.

References `jeod::ContactFacet::active`, `check_tree()`, `subject`, and `target`.

**8.4.3.9** `bool jeod::ContactPair::is_complete ( void )`

Determine if the pair has a target facet.

Returns

`bool`

Definition at line 100 of file `contact_pair.cc`.

References `target`.

**8.4.3.10** `ContactPair& jeod::ContactPair::operator= ( const ContactPair & rhs ) [private]`**8.4.4 Friends And Related Function Documentation****8.4.4.1** `void init_attrjeod__ContactPair ( ) [friend]`**8.4.4.2** `friend class InputProcessor [friend]`

Definition at line 51 of file `contact_pair.hh`.

**8.4.5 Field Documentation****8.4.5.1** `PairInteraction* jeod::ContactPair::interaction`

Parameters that define the force calculation function between the subject and target.

`trick_units(-)`

Definition at line 57 of file `contact_pair.hh`.

Referenced by `jeod::LineContactFacet::create_pair()`, `jeod::PointContactFacet::create_pair()`, `jeod::LineContactPair::in_contact()`, `jeod::PointContactPair::in_contact()`, and `jeod::LinePointContactPair::in_contact()`.

**8.4.5.2** `double jeod::ContactPair::interaction_distance`

`rel_state` distance at which `in_contact` should be called

`trick_units(m)`

Definition at line 62 of file `contact_pair.hh`.

Referenced by `jeod::LineContactFacet::create_pair()`, `jeod::PointContactFacet::create_pair()`, and `in_range()`.

#### 8.4.5.3 RelativeDerivedState jeod::ContactPair::rel\_state [protected]

Current relative state between the subject and the target in the subject frame.

trick\_units(-)

Definition at line 119 of file contact\_pair.hh.

Referenced by jeod::LineContactPair::in\_contact(), jeod::PointContactPair::in\_contact(), jeod::LinePointContactPair::in\_contact(), in\_range(), jeod::LineContactPair::initialize\_pair(), jeod::PointContactPair::initialize\_pair(), jeod::LinePointContactPair::initialize\_pair(), and initialize\_relstate().

#### 8.4.5.4 ContactFacet\* jeod::ContactPair::subject [protected]

pointer to the contact facet that is the subject of the associated relative states.

trick\_units(-)

Definition at line 124 of file contact\_pair.hh.

Referenced by check\_tree(), get\_subject(), jeod::LineContactPair::initialize\_pair(), jeod::PointContactPair::initialize\_pair(), jeod::LinePointContactPair::initialize\_pair(), initialize\_relstate(), and is\_active().

#### 8.4.5.5 ContactFacet\* jeod::ContactPair::target [protected]

pointer to the contact facet that is the target of the associated relative states.

trick\_units(-)

Definition at line 129 of file contact\_pair.hh.

Referenced by check\_tree(), get\_target(), jeod::LineContactPair::initialize\_pair(), jeod::PointContactPair::initialize\_pair(), jeod::LinePointContactPair::initialize\_pair(), is\_active(), and is\_complete().

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

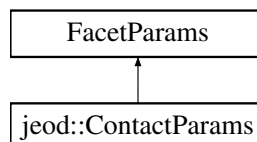
- [contact\\_pair.hh](#)
- [contact\\_pair.cc](#)

## 8.5 jeod::ContactParams Class Reference

A base class for all contact parameters used in the surface model.

```
#include <contact_params.hh>
```

Inheritance diagram for jeod::ContactParams:



### Public Member Functions

- [ContactParams](#) ()  
*Default Constructor.*
- virtual [~ContactParams](#) ()  
*Destructor.*



## Private Member Functions

- [ContactParams](#) & [operator=](#) (const [ContactParams](#) &rhs)
- [ContactParams](#) (const [ContactParams](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_ContactParams](#) ()

## 8.5.1 Detailed Description

A base class for all contact parameters used in the surface model.

Definition at line 46 of file [contact\\_params.hh](#).

## 8.5.2 Constructor & Destructor Documentation

### 8.5.2.1 `jeod::ContactParams::ContactParams ( void )`

Default Constructor.

Definition at line 43 of file [contact\\_params.cc](#).

### 8.5.2.2 `jeod::ContactParams::~~ContactParams ( void )` `[virtual]`

Destructor.

Definition at line 54 of file [contact\\_params.cc](#).

### 8.5.2.3 `jeod::ContactParams::ContactParams ( const ContactParams & rhs )` `[private]`

## 8.5.3 Member Function Documentation

### 8.5.3.1 `ContactParams& jeod::ContactParams::operator= ( const ContactParams & rhs )` `[private]`

## 8.5.4 Friends And Related Function Documentation

### 8.5.4.1 `void init_attrjeod__ContactParams ( )` `[friend]`

### 8.5.4.2 `friend class InputProcessor` `[friend]`

Definition at line 49 of file [contact\\_params.hh](#).

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

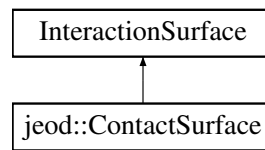
- [contact\\_params.hh](#)
- [contact\\_params.cc](#)

## 8.6 jeod::ContactSurface Class Reference

The contact specific interaction surface, for use with the surface model.

```
#include <contact_surface.hh>
```

Inheritance diagram for jeod::ContactSurface:



## Public Member Functions

- [ContactSurface](#) ()  
*Default Constructor.*
- virtual [~ContactSurface](#) ()  
*Destructor.*
- virtual void [allocate\\_array](#) (unsigned int size)  
*Allocates an array of [ContactFacet](#) pointers, of the size indicated by the input variable.*
- virtual void [allocate\\_interaction\\_facet](#) (Facet \*facet, InteractionFacetFactory \*factory, FacetParams \*params, unsigned int index)  
*Creates an interaction facet or more accurately a contact facet from a basic facet and set of parameters.*
- virtual void [collect\\_forces\\_torques](#) (void)  
*collect the forces and torques from all the facets in this contact surface*

## Data Fields

- [ContactFacet](#) \*\* [contact\\_facets](#)  
*An array of pointers to contact interaction facets.*
- unsigned int [facets\\_size](#)  
*Size of the [contact\\_facets](#) array.*
- double [contact\\_force](#) [3]  
*Total Force due to contact, resulting from all plates combined.*
- double [contact\\_torque](#) [3]  
*Total Torque due to contact, resulting from all plates combined.*

## Private Member Functions

- [ContactSurface](#) & [operator=](#) (const [ContactSurface](#) &rhs)
- [ContactSurface](#) (const [ContactSurface](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_ContactSurface](#) ()

### 8.6.1 Detailed Description

The contact specific interaction surface, for use with the surface model.

Definition at line 49 of file [contact\\_surface.hh](#).

## 8.6.2 Constructor & Destructor Documentation

### 8.6.2.1 jeod::ContactSurface::ContactSurface ( void )

Default Constructor.

Definition at line 59 of file contact\_surface.cc.

### 8.6.2.2 jeod::ContactSurface::~~ContactSurface ( void ) [virtual]

Destructor.

Definition at line 73 of file contact\_surface.cc.

References contact\_facets, and facets\_size.

### 8.6.2.3 jeod::ContactSurface::ContactSurface ( const ContactSurface & rhs ) [private]

## 8.6.3 Member Function Documentation

### 8.6.3.1 void jeod::ContactSurface::allocate\_array ( unsigned int size ) [virtual]

Allocates an array of [ContactFacet](#) pointers, of the size indicated by the input variable.

Parameters

in	size	The size of the needed array Units: cnt:
----	------	---

Definition at line 99 of file contact\_surface.cc.

References contact\_facets, facets\_size, and jeod::ContactMessages::initialization\_error.

### 8.6.3.2 void jeod::ContactSurface::allocate\_interaction\_facet ( Facet \* facet, InteractionFacetFactory \* factory, FacetParams \* params, unsigned int index ) [virtual]

Creates an interaction facet or more accurately a contact facet from a basic facet and set of parameters.

Parameters

in	facet	The basic facet used to create the interaction facet
in	factory	The factory used to create the interaction facet
in	params	The contact params used to create the interaction facet
in	index	Where the new interaction facet will be placed in the contact_facets array Units: cnt

Definition at line 137 of file contact\_surface.cc.

References contact\_facets, facets\_size, and jeod::ContactMessages::initialization\_error.

### 8.6.3.3 void jeod::ContactSurface::collect\_forces\_torques ( void ) [virtual]

collect the forces and torques from all the facets in this contact surface

Definition at line 214 of file contact\_surface.cc.

References contact\_facets, contact\_force, contact\_torque, and facets\_size.

### 8.6.3.4 ContactSurface& jeod::ContactSurface::operator= ( const ContactSurface & rhs ) [private]

## 8.6.4 Friends And Related Function Documentation

8.6.4.1 `void init_attrjeod__ContactSurface ( ) [friend]`

8.6.4.2 `friend class InputProcessor [friend]`

Definition at line 52 of file `contact_surface.hh`.

## 8.6.5 Field Documentation

8.6.5.1 `ContactFacet** jeod::ContactSurface::contact_facets`

An array of pointers to contact interaction facets.

`trick_units(-)`

Definition at line 65 of file `contact_surface.hh`.

Referenced by `allocate_array()`, `allocate_interaction_facet()`, `collect_forces_torques()`, and `~ContactSurface()`.

8.6.5.2 `double jeod::ContactSurface::contact_force[3]`

Total Force due to contact, resulting from all plates combined.

`trick_units(N)`

Definition at line 75 of file `contact_surface.hh`.

Referenced by `collect_forces_torques()`.

8.6.5.3 `double jeod::ContactSurface::contact_torque[3]`

Total Torque due to contact, resulting from all plates combined.

`trick_units(N/m)`

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

Referenced by `collect_forces_torques()`.

8.6.5.4 `unsigned int jeod::ContactSurface::facets_size`

Size of the `contact_facets` array.

`trick_units(count)`

Definition at line 70 of file `contact_surface.hh`.

Referenced by `allocate_array()`, `allocate_interaction_facet()`, `collect_forces_torques()`, and `~ContactSurface()`.

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

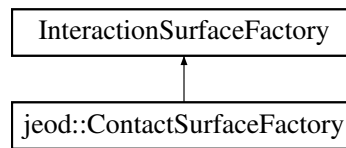
- [contact\\_surface.hh](#)
- [contact\\_surface.cc](#)

## 8.7 jeod::ContactSurfaceFactory Class Reference

The surface factory that creates an contact specific surface from a general surface.

```
#include <contact_surface_factory.hh>
```

Inheritance diagram for jeod::ContactSurfaceFactory:



## Public Member Functions

- [ContactSurfaceFactory](#) ()  
*Default Constructor.*
- virtual [~ContactSurfaceFactory](#) ()  
*Destructor.*
- virtual void [create\\_surface](#) (SurfaceModel \*surface, InteractionSurface \*inter\_surface)  
*Creates an interaction surface, in the inter\_surface parameter, from the given SurfaceModel.*
- virtual void [add\\_facet\\_params](#) (FacetParams \*to\_add)  
*Add a named set of facet params to the surface factory.*

## Protected Attributes

- [PointContactFacetFactory](#) point\_contact\_facet\_factory  
*A factory that can create a point contact facet from a circular flat plate.*
- [LineContactFacetFactory](#) line\_contact\_facet\_factory  
*A factory that can create a line contact facet from a cylinder.*

## Private Member Functions

- [ContactSurfaceFactory](#) & operator= (const [ContactSurfaceFactory](#) &rhs)
- [ContactSurfaceFactory](#) (const [ContactSurfaceFactory](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_ContactSurfaceFactory](#) ()

### 8.7.1 Detailed Description

The surface factory that creates an contact specific surface from a general surface.

Used with the surface model.

Definition at line 51 of file contact\_surface\_factory.hh.

### 8.7.2 Constructor & Destructor Documentation

#### 8.7.2.1 jeod::ContactSurfaceFactory::ContactSurfaceFactory ( void )

Default Constructor.

Definition at line 51 of file contact\_surface\_factory.cc.

References [line\\_contact\\_facet\\_factory](#), and [point\\_contact\\_facet\\_factory](#).

**8.7.2.2** `jeod::ContactSurfaceFactory::~~ContactSurfaceFactory ( void ) [virtual]`

Destructor.

Definition at line 65 of file `contact_surface_factory.cc`.

**8.7.2.3** `jeod::ContactSurfaceFactory::ContactSurfaceFactory ( const ContactSurfaceFactory & rhs ) [private]`

### 8.7.3 Member Function Documentation

**8.7.3.1** `void jeod::ContactSurfaceFactory::add_facet_params ( FacetParams * to_add ) [virtual]`

Add a named set of facet params to the surface factory.

Intended to be used when an contact specific surface is created, to convert a basic facet to an contact interaction facet. This MUST be a parameter inheriting from `ContactParam`, or the function will fail and send a failure message

Parameters

in	<i>to_add</i>	The facet parameters to add
----	---------------	-----------------------------

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

References `jeod::ContactMessages::pre_initialization_error`.

**8.7.3.2** `void jeod::ContactSurfaceFactory::create_surface ( SurfaceModel * surface, InteractionSurface * inter_surface ) [virtual]`

Creates an interaction surface, in the `inter_surface` parameter, from the given `SurfaceModel`.

The `InteractionSurfaceFactory` should contain all necessary `InteractionFacetFactories` and `FacetParams` already

Parameters

in	<i>surface</i>	The surface model used to create the interaction surface
out	<i>inter_surface</i>	Where the interaction surface will be produced

Definition at line 84 of file `contact_surface_factory.cc`.

References `jeod::ContactMessages::initialization_error`.

**8.7.3.3** `ContactSurfaceFactory& jeod::ContactSurfaceFactory::operator= ( const ContactSurfaceFactory & rhs ) [private]`

### 8.7.4 Friends And Related Function Documentation

**8.7.4.1** `void init_attrjeod__ContactSurfaceFactory ( ) [friend]`

**8.7.4.2** `friend class InputProcessor [friend]`

Definition at line 54 of file `contact_surface_factory.hh`.

### 8.7.5 Field Documentation

**8.7.5.1** `LineContactFacetFactory jeod::ContactSurfaceFactory::line_contact_facet_factory [protected]`

A factory that can create a line contact facet from a cylinder.

`trick_units(-)`

Definition at line 87 of file `contact_surface_factory.hh`.

Referenced by `ContactSurfaceFactory()`.

#### 8.7.5.2 PointContactFacetFactory jeod::ContactSurfaceFactory::point\_contact\_facet\_factory [protected]

A factory that can create a point contact facet from a circular flat plate.

`trick_units(-)`

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

Referenced by `ContactSurfaceFactory()`.

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

- [contact\\_surface\\_factory.hh](#)
- [contact\\_surface\\_factory.cc](#)

## 8.8 jeod::ContactUtils Class Reference

Utility string and math functions for the contact model.

```
#include <contact_utils.hh>
```

### Static Public Member Functions

- static int [create\\_relstate\\_name](#) (char \*name1, char \*name2, char \*\*out\_str)  
*create a name for a relstate out of two facet names*
- static int [copy\\_const\\_char\\_to\\_char](#) (const char \*in\_str, char \*\*out\_str)  
*create a name for a relstate out of two facet names*
- static int [dist\\_line\\_segments](#) (double p1[3], double p2[3], double p3[3], double p4[3], double \*pa, double \*pb)  
*calculate the closest points between two line segments*

#### 8.8.1 Detailed Description

Utility string and math functions for the contact model.

Definition at line 36 of file `contact_utils.hh`.

#### 8.8.2 Member Function Documentation

##### 8.8.2.1 int jeod::ContactUtils::copy\_const\_char\_to\_char ( const char \* in\_str, char \*\* out\_str ) [inline], [static]

create a name for a relstate out of two facet names

**Returns**

int

**Parameters**

---

in	<i>in_str</i>	const char input
in, out	<i>out_str</i>	char output

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

Referenced by `jeod::LineContactPair::initialize_pair()`, `jeod::PointContactPair::initialize_pair()`, and `jeod::LinePointContactPair::initialize_pair()`.

**8.8.2.2** `int jeod::ContactUtils::create_relstate_name ( char * name1, char * name2, char ** out_str ) [inline], [static]`

create a name for a relstate out of two facet names

#### Returns

char\*\*

#### Parameters

in	<i>name1</i>	name of first contact facet
in	<i>name2</i>	name of second contact facet
in, out	<i>out_str</i>	output name for the relstate

Definition at line 60 of file `contact_utils_inline.hh`.

**8.8.2.3** `int jeod::ContactUtils::dist_line_segments ( double p1[3], double p2[3], double p3[3], double p4[3], double * pa, double * pb ) [inline], [static]`

calculate the closest points between two line segments

#### Returns

success

#### Parameters

in	<i>p1</i>	vector to one point of the first line seg Units: M
in	<i>p2</i>	vector to one point of the first line seg Units: M
in	<i>p3</i>	vector to one point of the second line seg Units: M
in	<i>p4</i>	vector to one point of the second line seg Units: M
out	<i>pa</i>	vector to close_point on line 1 Units: M
out	<i>pb</i>	vector to close_point on line 2 Units: M

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

Referenced by `jeod::LineContactPair::in_contact()`, and `jeod::LinePointContactPair::in_contact()`.

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

- [contact\\_utils.hh](#)
- [contact\\_utils\\_inline.hh](#)

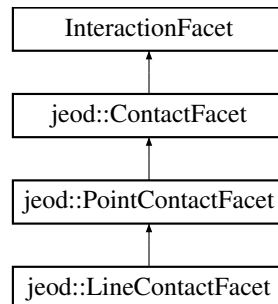


## 8.9 jeod::LineContactFacet Class Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

```
#include <line_contact_facet.hh>
```

Inheritance diagram for jeod::LineContactFacet:



### Public Member Functions

- [LineContactFacet](#) ()  
*Default constructor.*
- virtual [~LineContactFacet](#) ()  
*Destructor.*
- virtual [ContactPair](#) \* [create\\_pair](#) ()  
*Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*
- virtual [ContactPair](#) \* [create\\_pair](#) ([ContactFacet](#) \*target, [Contact](#) \*contact)  
*Overloaded functions that creates a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*
- virtual void [set\\_max\\_dimension](#) ()  
*calculate the max dimension of the facet for range limit determination.*
- virtual void [calculate\\_torque](#) (double \*tmp\_force)  
*Calculate the torque generated on the vehicle by the facet.*
- virtual void [calculate\\_contact\\_point](#) (double nvec[3])  
*Find the point on the surface that coorisponds to the closest point on line segments using the radius value.*

### Data Fields

- double [length](#)  
*length of the line along the vehicle point x axis.*

### Private Member Functions

- [LineContactFacet](#) & [operator=](#) (const [LineContactFacet](#) &rhs)
- [LineContactFacet](#) (const [LineContactFacet](#) &rhs)

### Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_LineContactFacet](#) ()

### 8.9.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point.

In effect this represents a sphere.

Definition at line 51 of file line\_contact\_facet.hh.

### 8.9.2 Constructor & Destructor Documentation

#### 8.9.2.1 jeod::LineContactFacet::LineContactFacet ( void )

Default constructor.

Definition at line 53 of file line\_contact\_facet.cc.

References jeod::PointContactFacet::contact\_point.

#### 8.9.2.2 jeod::LineContactFacet::~~LineContactFacet ( void ) [virtual]

Destructor.

Definition at line 67 of file line\_contact\_facet.cc.

#### 8.9.2.3 jeod::LineContactFacet::LineContactFacet ( const LineContactFacet & rhs ) [private]

### 8.9.3 Member Function Documentation

#### 8.9.3.1 void jeod::LineContactFacet::calculate\_contact\_point ( double nvec[3] ) [virtual]

Find the point on the surface that corresponds to the closest point on line segments using the radius value.

Parameters

in	<i>nvec</i>	vector between line points
----	-------------	----------------------------

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

Definition at line 173 of file line\_contact\_facet.cc.

References jeod::PointContactFacet::contact\_point, length, and jeod::PointContactFacet::radius.

Referenced by jeod::LineContactPair::in\_contact(), and jeod::LinePointContactPair::in\_contact().

#### 8.9.3.2 void jeod::LineContactFacet::calculate\_torque ( double \* tmp\_force ) [virtual]

Calculate the torque generated on the vehicle by the facet.

Assumes that the force is in the vehicle structural frame, but that close\_point is not.

Parameters

in	<i>tmp_force</i>	force from one contact interaction. Units: N
----	------------------	---

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

Definition at line 231 of file line\_contact\_facet.cc.

References jeod::PointContactFacet::contact\_point, jeod::ContactFacet::vehicle\_body, and jeod::ContactFacet::vehicle\_point.

**8.9.3.3** `ContactPair * jeod::LineContactFacet::create_pair ( void )` [virtual]

Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

**Returns**

[ContactPair](#) that was created

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

Definition at line 83 of file `line_contact_facet.cc`.

References [jeod::LineContactPair::initialize\\_pair\(\)](#).

**8.9.3.4** `ContactPair * jeod::LineContactFacet::create_pair ( ContactFacet * target, Contact * contact )` [virtual]

Overloaded functions that creates a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.

This function is called when a subject and target are known.

**Returns**

[ContactPair](#) that was created

**Parameters**

<code>in, out</code>	<code>target</code>	target <a href="#">ContactFacet</a>
<code>in</code>	<code>contact</code>	<a href="#">Contact</a> object used to find the pair interaction

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

Definition at line 104 of file `line_contact_facet.cc`.

References [jeod::Contact::contact\\_limit\\_factor](#), [jeod::Contact::find\\_interaction\(\)](#), [jeod::ContactMessages::initialization\\_warns](#), [jeod::LineContactPair::initialize\\_pair\(\)](#), [jeod::LinePointContactPair::initialize\\_pair\(\)](#), [jeod::ContactPair::interaction](#), [jeod::ContactPair::interaction\\_distance](#), [jeod::ContactFacet::max\\_dimension](#), and [jeod::ContactFacet::surface\\_type](#).

**8.9.3.5** `LineContactFacet& jeod::LineContactFacet::operator= ( const LineContactFacet & rhs )` [private]**8.9.3.6** `void jeod::LineContactFacet::set_max_dimension ( void )` [virtual]

calculate the max dimension of the facet for range limit determination.

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

Definition at line 218 of file `line_contact_facet.cc`.

References [length](#), [jeod::ContactFacet::max\\_dimension](#), and [jeod::PointContactFacet::radius](#).

Referenced by [jeod::LineContactFacetFactory::create\\_facet\(\)](#).

**8.9.4 Friends And Related Function Documentation****8.9.4.1** `void init_attrjeod_LineContactFacet ( )` [friend]**8.9.4.2** `friend class InputProcessor` [friend]

Definition at line 53 of file `line_contact_facet.hh`.

## 8.9.5 Field Documentation

### 8.9.5.1 double jeod::LineContactFacet::length

length of the line along the vehicle point x axis.

trick\_units(m)

Definition at line 59 of file line\_contact\_facet.hh.

Referenced by calculate\_contact\_point(), jeod::LineContactFacetFactory::create\_facet(), jeod::LineContactPair::in\_contact(), jeod::LinePointContactPair::in\_contact(), and set\_max\_dimension().

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

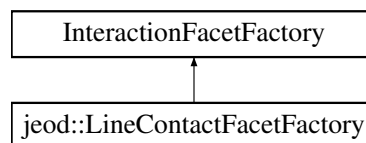
- [line\\_contact\\_facet.hh](#)
- [line\\_contact\\_facet.cc](#)

## 8.10 jeod::LineContactFacetFactory Class Reference

Creates a [PointContactFacet](#) from an InteractionFacet.

```
#include <line_contact_facet_factory.hh>
```

Inheritance diagram for jeod::LineContactFacetFactory:



### Public Member Functions

- [LineContactFacetFactory](#) ()  
*Default Constructor.*
- [~LineContactFacetFactory](#) ()  
*Destructor.*
- virtual InteractionFacet \* [create\\_facet](#) (Facet \*facet, FacetParams \*params)  
*Create a [PointContactFacet](#) from a CircularFlatPlate facet and a [ContactParams](#) object.*
- virtual bool [is\\_correct\\_factory](#) (Facet \*facet)  
*[PointContactFacetFactory](#) specific implementation of this function.*

### Private Member Functions

- [LineContactFacetFactory](#) & operator= (const [LineContactFacetFactory](#) &rhs)
- [LineContactFacetFactory](#) (const [LineContactFacetFactory](#) &rhs)

### Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_LineContactFacetFactory](#) ()

### 8.10.1 Detailed Description

Creates a [PointContactFacet](#) from an InteractionFacet.

Definition at line 50 of file line\_contact\_facet\_factory.hh.

### 8.10.2 Constructor & Destructor Documentation

#### 8.10.2.1 jeod::LineContactFacetFactory::LineContactFacetFactory ( void )

Default Constructor.

Definition at line 54 of file line\_contact\_facet\_factory.cc.

#### 8.10.2.2 jeod::LineContactFacetFactory::~~LineContactFacetFactory ( void )

Destructor.

Definition at line 65 of file line\_contact\_facet\_factory.cc.

#### 8.10.2.3 jeod::LineContactFacetFactory::LineContactFacetFactory ( const LineContactFacetFactory & rhs ) [private]

### 8.10.3 Member Function Documentation

#### 8.10.3.1 InteractionFacet \* jeod::LineContactFacetFactory::create\_facet ( Facet \* facet, FacetParams \* params ) [virtual]

Create a [PointContactFacet](#) from a CircularFlatPlate facet and a [ContactParams](#) object.

##### Returns

The new EllipsoidContactFacet. Note that this is allocated and YOU are responsible for destroying it at the end!

##### Parameters

in	<i>facet</i>	The CircularFlatPlate. This MUST be a circular flat plate or the algorithm will send a failure message
in	<i>params</i>	<a href="#">ContactParams</a>

Definition at line 83 of file line\_contact\_facet\_factory.cc.

References [jeod::ContactFacet::create\\_vehicle\\_point\(\)](#), [jeod::ContactMessages::initialization\\_error](#), [jeod::LineContactFacet::length](#), [jeod::ContactFacet::normal](#), [jeod::ContactFacet::position](#), [jeod::PointContactFacet::radius](#), [jeod::LineContactFacet::set\\_max\\_dimension\(\)](#), [jeod::ContactFacet::surface\\_type](#), and [jeod::ContactFacet::vehicle\\_body](#).

#### 8.10.3.2 bool jeod::LineContactFacetFactory::is\_correct\_factory ( Facet \* facet ) [virtual]

[PointContactFacetFactory](#) specific implementation of this function.

If the Facet is of type CircularFlatPlate, returns true. False otherwise

##### Returns

true if facet is a FlatPlateCircular, false otherwise

## Parameters

<code>in</code>	<code>facet</code>	The facet to check
-----------------	--------------------	--------------------

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

**8.10.3.3** `LineContactFacetFactory& jeod::LineContactFacetFactory::operator= ( const LineContactFacetFactory & rhs ) [private]`

## 8.10.4 Friends And Related Function Documentation

**8.10.4.1** `void init_attrjeod__LineContactFacetFactory ( ) [friend]`

**8.10.4.2** `friend class InputProcessor [friend]`

Definition at line 53 of file `line_contact_facet_factory.hh`.

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

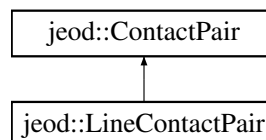
- [line\\_contact\\_facet\\_factory.hh](#)
- [line\\_contact\\_facet\\_factory.cc](#)

## 8.11 jeod::LineContactPair Class Reference

An point to point contact pair for use in the contact model.

```
#include <line_contact_pair.hh>
```

Inheritance diagram for `jeod::LineContactPair`:



### Public Member Functions

- [LineContactPair](#) ()  
*Default Constructor.*
- virtual [~LineContactPair](#) ()  
*Destructor.*
- virtual void [in\\_contact](#) ()  
*Determine if contact has occurred between the facets of the pair.*
- virtual void [initialize\\_pair](#) ([ContactFacet](#) \*subject\_facet, [ContactFacet](#) \*target\_facet)  
*Initialize the contact pair by setting the subject, target, and creating the relstate if possible.*

### Data Fields

- [LineContactFacet](#) \* [line\\_subject](#)  
*pointer to the contact facet that is the subject of the associated relative states.*
- [LineContactFacet](#) \* [line\\_target](#)  
*pointer to the contact facet that is the target of the associated relative states.*

## Private Member Functions

- [LineContactPair](#) & [operator=](#) (const [LineContactPair](#) &rhs)
- [LineContactPair](#) (const [LineContactPair](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_LineContactPair](#) ()

## Additional Inherited Members

### 8.11.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 48 of file `line_contact_pair.hh`.

### 8.11.2 Constructor & Destructor Documentation

#### 8.11.2.1 jeod::LineContactPair::LineContactPair ( void )

Default Constructor.

Definition at line 48 of file `line_contact_pair.cc`.

#### 8.11.2.2 jeod::LineContactPair::~~LineContactPair ( void ) [virtual]

Destructor.

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

#### 8.11.2.3 jeod::LineContactPair::LineContactPair ( const LineContactPair & rhs ) [private]

### 8.11.3 Member Function Documentation

#### 8.11.3.1 void jeod::LineContactPair::in\_contact ( void ) [virtual]

Determine if contact has occurred between the facets of the pair.

Implements [jeod::ContactPair](#).

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

References [jeod::LineContactFacet::calculate\\_contact\\_point\(\)](#), [jeod::PairInteraction::calculate\\_forces\(\)](#), [jeod::PointContactFacet::contact\\_point](#), [jeod::ContactUtils::dist\\_line\\_segments\(\)](#), [jeod::ContactPair::interaction](#), [jeod::LineContactFacet::length](#), [line\\_subject](#), [line\\_target](#), and [jeod::ContactPair::rel\\_state](#).

#### 8.11.3.2 void jeod::LineContactPair::initialize\_pair ( ContactFacet \* subject\_facet, ContactFacet \* target\_facet ) [virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

## Parameters

<i>in, out</i>	<i>subject_facet</i>	subject <a href="#">ContactFacet</a>
<i>in, out</i>	<i>target_facet</i>	target <a href="#">ContactFacet</a>

Implements [jeod::ContactPair](#).

Definition at line 173 of file `line_contact_pair.cc`.

References `jeod::ContactUtils::copy_const_char_to_char()`, `line_subject`, `line_target`, `jeod::ContactPair::rel_state`, `jeod::ContactPair::subject`, `jeod::ContactPair::target`, and `jeod::ContactFacet::vehicle_point`.

Referenced by `jeod::LineContactFacet::create_pair()`.

**8.11.3.3** `LineContactPair& jeod::LineContactPair::operator= ( const LineContactPair & rhs )` `[private]`

## 8.11.4 Friends And Related Function Documentation

**8.11.4.1** `void init_attrjeod__LineContactPair ( )` `[friend]`

**8.11.4.2** `friend class InputProcessor` `[friend]`

Definition at line 50 of file `line_contact_pair.hh`.

## 8.11.5 Field Documentation

**8.11.5.1** `LineContactFacet* jeod::LineContactPair::line_subject`

pointer to the contact facet that is the subject of the associated relative states.

`trick_units(-)`

Definition at line 56 of file `line_contact_pair.hh`.

Referenced by `in_contact()`, and `initialize_pair()`.

**8.11.5.2** `LineContactFacet* jeod::LineContactPair::line_target`

pointer to the contact facet that is the target of the associated relative states.

`trick_units(-)`

Definition at line 61 of file `line_contact_pair.hh`.

Referenced by `in_contact()`, and `initialize_pair()`.

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

- [line\\_contact\\_pair.hh](#)
- [line\\_contact\\_pair.cc](#)

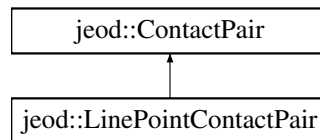
## 8.12 jeod::LinePointContactPair Class Reference

An point to point contact pair for use in the contact model.

`#include <line_point_contact_pair.hh>`

Inheritance diagram for `jeod::LinePointContactPair`:





## Public Member Functions

- [LinePointContactPair](#) ()  
*Default Constructor.*
- virtual [~LinePointContactPair](#) ()  
*Destructor.*
- virtual void [in\\_contact](#) ()  
*Determine if contact has occurred between the facets of the pair.*
- virtual void [initialize\\_pair](#) ([ContactFacet](#) \*subject\_facet, [ContactFacet](#) \*target\_facet)  
*Initialize the contact pair by setting the subject, target, and creating the relstate if possible.*

## Data Fields

- [LineContactFacet](#) \* [line\\_subject](#)  
*pointer to the contact facet that is the subject of the associated relative states.*
- [PointContactFacet](#) \* [point\\_target](#)  
*pointer to the contact facet that is the target of the associated relative states.*

## Private Member Functions

- [LinePointContactPair](#) & [operator=](#) (const [LineContactPair](#) &rhs)
- [LinePointContactPair](#) (const [LineContactPair](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_LinePointContactPair](#) ()

## Additional Inherited Members

### 8.12.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 49 of file `line_point_contact_pair.hh`.

### 8.12.2 Constructor & Destructor Documentation

#### 8.12.2.1 jeod::LinePointContactPair::LinePointContactPair ( void )

Default Constructor.

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

8.12.2.2 `jeod::LinePointContactPair::~~LinePointContactPair ( void ) [virtual]`

Destructor.

Definition at line 62 of file `line_point_contact_pair.cc`.

8.12.2.3 `jeod::LinePointContactPair::LinePointContactPair ( const LineContactPair & rhs ) [private]`

### 8.12.3 Member Function Documentation

8.12.3.1 `void jeod::LinePointContactPair::in_contact ( void ) [virtual]`

Determine if contact has occurred between the facets of the pair.

Implements [jeod::ContactPair](#).

Definition at line 72 of file `line_point_contact_pair.cc`.

References `jeod::LineContactFacet::calculate_contact_point()`, `jeod::PointContactFacet::calculate_contact_point()`, `jeod::PairInteraction::calculate_forces()`, `jeod::PointContactFacet::contact_point`, `jeod::ContactUtils::dist_line_segments()`, `jeod::ContactPair::interaction`, `jeod::LineContactFacet::length`, `line_subject`, `point_target`, and `jeod::ContactPair::rel_state`.

8.12.3.2 `void jeod::LinePointContactPair::initialize_pair ( ContactFacet * subject_facet, ContactFacet * target_facet ) [virtual]`

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

<code>in, out</code>	<code>subject_facet</code>	subject <a href="#">ContactFacet</a>
<code>in, out</code>	<code>target_facet</code>	target <a href="#">ContactFacet</a>

Implements [jeod::ContactPair](#).

Definition at line 158 of file `line_point_contact_pair.cc`.

References `jeod::ContactUtils::copy_const_char_to_char()`, `line_subject`, `point_target`, `jeod::ContactPair::rel_state`, `jeod::ContactPair::subject`, `jeod::ContactPair::target`, and `jeod::ContactFacet::vehicle_point`.

Referenced by `jeod::LineContactFacet::create_pair()`.

8.12.3.3 `LinePointContactPair& jeod::LinePointContactPair::operator= ( const LineContactPair & rhs ) [private]`

### 8.12.4 Friends And Related Function Documentation

8.12.4.1 `void init_attrjeod__LinePointContactPair ( ) [friend]`

8.12.4.2 `friend class InputProcessor [friend]`

Definition at line 51 of file `line_point_contact_pair.hh`.

### 8.12.5 Field Documentation

8.12.5.1 `LineContactFacet* jeod::LinePointContactPair::line_subject`

pointer to the contact facet that is the subject of the associated relative states.

`trick_units(-)`

Definition at line 57 of file line\_point\_contact\_pair.hh.

Referenced by in\_contact(), and initialize\_pair().

#### 8.12.5.2 PointContactFacet\* jeod::LinePointContactPair::point\_target

pointer to the contact facet that is the target of the associated relative states.

trick\_units(-)

Definition at line 62 of file line\_point\_contact\_pair.hh.

Referenced by in\_contact(), and initialize\_pair().

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

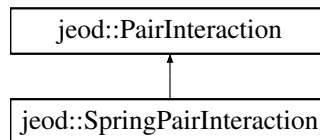
- [line\\_point\\_contact\\_pair.hh](#)
- [line\\_point\\_contact\\_pair.cc](#)

## 8.13 jeod::PairInteraction Class Reference

Simple spring contact parameters.

```
#include <pair_interaction.hh>
```

Inheritance diagram for jeod::PairInteraction:



### Public Member Functions

- [PairInteraction](#) ()  
*Default Constructor.*
- virtual [~PairInteraction](#) ()  
*Destructor.*
- bool [is\\_correct\\_interaction](#) (ContactParams \*subject\_params, ContactParams \*target\_params)  
*Check a pair of contact params for a match to the ones defined for this pair\_interaction.*
- virtual void [calculate\\_forces](#) (ContactFacet \*subject, ContactFacet \*target, RelativeDerivedState \*rel\_state, double \*penetration\_vector, double \*rel\_velocity)=0  
*Pure virtual function that is defined to calculate forces on facets in contact.*

### Data Fields

- char \* [params\\_1](#)  
*contact param type that defines this pair interaction.*
- char \* [params\\_2](#)  
*contact param type that defines this pair interaction.*
- double [friction\\_mag](#)  
*magnitude of the friction force on the contact surfaces.*

## Private Member Functions

- [PairInteraction](#) & `operator=` (const [PairInteraction](#) &rhs)
- [PairInteraction](#) (const [PairInteraction](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_PairInteraction](#) ()

### 8.13.1 Detailed Description

Simple spring contact parameters.

Definition at line 50 of file `pair_interaction.hh`.

### 8.13.2 Constructor & Destructor Documentation

#### 8.13.2.1 `jeod::PairInteraction::PairInteraction ( void )`

Default Constructor.

Definition at line 47 of file `pair_interaction.cc`.

#### 8.13.2.2 `jeod::PairInteraction::~~PairInteraction ( void ) [virtual]`

Destructor.

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

#### 8.13.2.3 `jeod::PairInteraction::PairInteraction ( const PairInteraction & rhs ) [private]`

### 8.13.3 Member Function Documentation

#### 8.13.3.1 `virtual void jeod::PairInteraction::calculate_forces ( ContactFacet * subject, ContactFacet * target, RelativeDerivedState * rel_state, double * penetration_vector, double * rel_velocity ) [pure virtual]`

Pure virtual function that is defined to calculate forces on facets in contact.

##### Parameters

<code>in, out</code>	<i>subject</i>	subject of the relative state
<code>in, out</code>	<i>target</i>	target of the relative state
<code>in</code>	<i>rel_state</i>	relative state between subject and target in subject frame
<code>in</code>	<i>penetration_vector</i>	vector that characterises the interpenetration of the subject and the target
<code>in</code>	<i>rel_velocity</i>	relative velocity of the subject and the target in the subject frame

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

Referenced by `jeod::LineContactPair::in_contact()`, `jeod::PointContactPair::in_contact()`, and `jeod::LinePointContactPair::in_contact()`.

#### 8.13.3.2 `bool jeod::PairInteraction::is_correct_interaction ( ContactParams * subject_params, ContactParams * target_params )`

Check a pair of contact params for a match to the ones defined for this `pair_interaction`.

**Returns**

bool

**Parameters**

in	<i>subject_params</i>	parameters of the subject
in	<i>target_params</i>	parameters of the target

Definition at line 75 of file pair\_interaction.cc.

References `params_1`, and `params_2`.

**8.13.3.3** `PairInteraction& jeod::PairInteraction::operator= ( const PairInteraction & rhs )` `[private]`

**8.13.4 Friends And Related Function Documentation**

**8.13.4.1** `void init_attrjeod_PairInteraction ( )` `[friend]`

**8.13.4.2** `friend class InputProcessor` `[friend]`

Definition at line 52 of file pair\_interaction.hh.

**8.13.5 Field Documentation**

**8.13.5.1** `double jeod::PairInteraction::friction_mag`

magnitude of the friction force on the contact surfaces.

trick\_units(N)

Definition at line 68 of file pair\_interaction.hh.

Referenced by `jeod::SpringPairInteraction::calculate_forces()`.

**8.13.5.2** `char* jeod::PairInteraction::params_1`

contact param type that defines this pair interaction.

trick\_units(-)

Definition at line 59 of file pair\_interaction.hh.

Referenced by `is_correct_interaction()`.

**8.13.5.3** `char* jeod::PairInteraction::params_2`

contact param type that defines this pair interaction.

trick\_units(-)

Definition at line 63 of file pair\_interaction.hh.

Referenced by `is_correct_interaction()`.

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

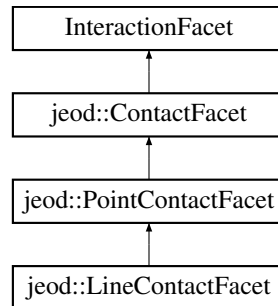
- [pair\\_interaction.hh](#)
- [pair\\_interaction.cc](#)

## 8.14 jeod::PointContactFacet Class Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

```
#include <point_contact_facet.hh>
```

Inheritance diagram for jeod::PointContactFacet:



### Public Member Functions

- [PointContactFacet](#) ()  
*Default constructor.*
- virtual [~PointContactFacet](#) ()  
*Destructor.*
- virtual [ContactPair](#) \* [create\\_pair](#) ()  
*Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*
- virtual [ContactPair](#) \* [create\\_pair](#) ([ContactFacet](#) \*target, [Contact](#) \*contact)  
*Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.*
- virtual void [set\\_max\\_dimension](#) ()  
*calculate the max dimension of the facet for range limit determination.*
- virtual void [calculate\\_contact\\_point](#) (double nvec[3])  
*Use the relstate and radius of contact to calculate a contact point on this facet.*
- virtual void [calculate\\_torque](#) (double \*tmp\_force)  
*Calculate the torque generated on the vehicle by the facet.*

### Data Fields

- double [radius](#)  
*radius from the point at which contact takes place.*
- double [contact\\_point](#) [3]  
*[Contact](#) point given in facet vehicle point frame, representing point on the point on the surface of a sphere of radius "radius" where contact has occurred.*

### Private Member Functions

- [PointContactFacet](#) & [operator=](#) (const [PointContactFacet](#) &rhs)
- [PointContactFacet](#) (const [PointContactFacet](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_PointContactFacet](#) ()

### 8.14.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point.

In effect this represents a sphere.

Definition at line 51 of file `point_contact_facet.hh`.

### 8.14.2 Constructor & Destructor Documentation

#### 8.14.2.1 jeod::PointContactFacet::PointContactFacet ( void )

Default constructor.

Definition at line 51 of file `point_contact_facet.cc`.

References `contact_point`.

#### 8.14.2.2 jeod::PointContactFacet::~~PointContactFacet ( void ) [virtual]

Destructor.

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

#### 8.14.2.3 jeod::PointContactFacet::PointContactFacet ( const PointContactFacet & rhs ) [private]

### 8.14.3 Member Function Documentation

#### 8.14.3.1 void jeod::PointContactFacet::calculate\_contact\_point ( double nvec[3] ) [virtual]

Use the relstate and radius of contact to calculate a contact point on this facet.

Parameters

<code>in</code>	<code>nvec</code>	direction vector between the two facets
-----------------	-------------------	---

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

Definition at line 139 of file `point_contact_facet.cc`.

References `contact_point`, and `radius`.

Referenced by `jeod::PointContactPair::in_contact()`, and `jeod::LinePointContactPair::in_contact()`.

#### 8.14.3.2 void jeod::PointContactFacet::calculate\_torque ( double \* tmp\_force ) [virtual]

Calculate the torque generated on the vehicle by the facet.

Assumes that the force is in the vehicle structural frame, but that `close_point` is not.

Parameters

<i>in</i>	<i>tmp_force</i>	force from one contact interaction. Units: N
-----------	------------------	---

Implements [jeod::ContactFacet](#).

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

Definition at line 167 of file point\_contact\_facet.cc.

References [contact\\_point](#), [jeod::ContactFacet::vehicle\\_body](#), and [jeod::ContactFacet::vehicle\\_point](#).

#### 8.14.3.3 **ContactPair \* jeod::PointContactFacet::create\_pair ( void )** [virtual]

Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

Returns

[ContactPair](#) that was created

Implements [jeod::ContactFacet](#).

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

Definition at line 80 of file point\_contact\_facet.cc.

References [jeod::PointContactPair::initialize\\_pair\(\)](#).

#### 8.14.3.4 **ContactPair \* jeod::PointContactFacet::create\_pair ( ContactFacet \* target, Contact \* contact )** [virtual]

Overloaded functions that create a [ContactPair](#) and pass the address of it to the [Contact](#) class for addition to the list of pairs.

This function is called when a subject and target are known.

Returns

[ContactPair](#) that was created

Parameters

<i>in, out</i>	<i>target</i>	target <a href="#">ContactFacet</a>
<i>in</i>	<i>contact</i>	<a href="#">Contact</a> object used to find the pair interaction

Implements [jeod::ContactFacet](#).

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

Definition at line 101 of file point\_contact\_facet.cc.

References [jeod::Contact::contact\\_limit\\_factor](#), [jeod::Contact::find\\_interaction\(\)](#), [jeod::ContactMessages::initialization\\_warns](#), [jeod::PointContactPair::initialize\\_pair\(\)](#), [jeod::ContactPair::interaction](#), [jeod::ContactPair::interaction\\_distance](#), [jeod::ContactFacet::max\\_dimension](#), and [jeod::ContactFacet::surface\\_type](#).

#### 8.14.3.5 **PointContactFacet& jeod::PointContactFacet::operator= ( const PointContactFacet & rhs )** [private]

#### 8.14.3.6 **void jeod::PointContactFacet::set\_max\_dimension ( void )** [virtual]

calculate the max dimension of the facet for range limit determination.

Implements [jeod::ContactFacet](#).



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

Definition at line 154 of file `point_contact_facet.cc`.

References `jeod::ContactFacet::max_dimension`, and `radius`.

Referenced by `jeod::PointContactFacetFactory::create_facet()`.

## 8.14.4 Friends And Related Function Documentation

8.14.4.1 `void init_attrjeod__PointContactFacet( ) [friend]`

8.14.4.2 `friend class InputProcessor [friend]`

Definition at line 53 of file `point_contact_facet.hh`.

## 8.14.5 Field Documentation

8.14.5.1 `double jeod::PointContactFacet::contact_point[3]`

[Contact](#) point given in facet vehicle point frame, representing point on the point on the surface of a sphere of radius "radius" where contact has occurred.

`trick_units(m)`

Definition at line 66 of file `point_contact_facet.hh`.

Referenced by `jeod::LineContactFacet::calculate_contact_point()`, `calculate_contact_point()`, `jeod::LineContactFacet::calculate_torque()`, `calculate_torque()`, `jeod::LineContactPair::in_contact()`, `jeod::PointContactPair::in_contact()`, `jeod::LinePointContactPair::in_contact()`, `jeod::LineContactFacet::LineContactFacet()`, and `PointContactFacet()`.

8.14.5.2 `double jeod::PointContactFacet::radius`

radius from the point at which contact takes place.

`trick_units(m)`

Definition at line 59 of file `point_contact_facet.hh`.

Referenced by `jeod::LineContactFacet::calculate_contact_point()`, `calculate_contact_point()`, `jeod::PointContactFacetFactory::create_facet()`, `jeod::LineContactFacetFactory::create_facet()`, `jeod::PointContactPair::in_contact()`, `jeod::LineContactFacet::set_max_dimension()`, and `set_max_dimension()`.

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

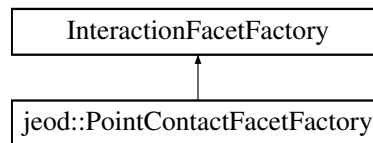
- [point\\_contact\\_facet.hh](#)
- [point\\_contact\\_facet.cc](#)

## 8.15 jeod::PointContactFacetFactory Class Reference

Creates a [PointContactFacet](#) from an `InteractionFacet`.

```
#include <point_contact_facet_factory.hh>
```

Inheritance diagram for `jeod::PointContactFacetFactory`:



## Public Member Functions

- [PointContactFacetFactory](#) ()  
*Default Constructor.*
- [~PointContactFacetFactory](#) ()  
*Destructor.*
- virtual InteractionFacet \* [create\\_facet](#) (Facet \*facet, FacetParams \*params)  
*Create a [PointContactFacet](#) from a CircularFlatPlate facet and a [ContactParams](#) object.*
- virtual bool [is\\_correct\\_factory](#) (Facet \*facet)  
*[PointContactFacetFactory](#) specific implementation of this function.*

## Private Member Functions

- [PointContactFacetFactory](#) & [operator=](#) (const [PointContactFacetFactory](#) &rhs)
- [PointContactFacetFactory](#) (const [PointContactFacetFactory](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_PointContactFacetFactory](#) ()

### 8.15.1 Detailed Description

Creates a [PointContactFacet](#) from an InteractionFacet.

Definition at line 50 of file point\_contact\_facet\_factory.hh.

### 8.15.2 Constructor & Destructor Documentation

#### 8.15.2.1 jeod::PointContactFacetFactory::PointContactFacetFactory ( void )

Default Constructor.

Definition at line 54 of file point\_contact\_facet\_factory.cc.

#### 8.15.2.2 jeod::PointContactFacetFactory::~~PointContactFacetFactory ( void )

Destructor.

Definition at line 65 of file point\_contact\_facet\_factory.cc.

8.15.2.3 `jeod::PointContactFacetFactory::PointContactFacetFactory ( const PointContactFacetFactory & rhs )`  
`[private]`

### 8.15.3 Member Function Documentation

8.15.3.1 `InteractionFacet * jeod::PointContactFacetFactory::create_facet ( Facet * facet, FacetParams * params )`  
`[virtual]`

Create a [PointContactFacet](#) from a [CircularFlatPlate](#) facet and a [ContactParams](#) object.

#### Returns

The new [EllipsoidContactFacet](#). Note that this is allocated and YOU are responsible for destroying it at the end!

#### Parameters

in	<i>facet</i>	The <a href="#">CircularFlatPlate</a> . This MUST be a circular flat plate or the algorithm will send a failure message
in	<i>params</i>	<a href="#">ContactParams</a>

Definition at line 83 of file `point_contact_facet_factory.cc`.

References `jeod::ContactFacet::create_vehicle_point()`, `jeod::ContactMessages::initialization_error`, `jeod::ContactFacet::normal`, `jeod::ContactFacet::position`, `jeod::PointContactFacet::radius`, `jeod::PointContactFacet::set_max_dimension()`, `jeod::ContactFacet::surface_type`, and `jeod::ContactFacet::vehicle_body`.

8.15.3.2 `bool jeod::PointContactFacetFactory::is_correct_factory ( Facet * facet )` `[virtual]`

[PointContactFacetFactory](#) specific implementation of this function.

If the Facet is of type [CircularFlatPlate](#), returns true. False otherwise

#### Returns

true if facet is a [FlatPlateCircular](#), false otherwise

#### Parameters

in	<i>facet</i>	The facet to check
----	--------------	--------------------

Definition at line 160 of file `point_contact_facet_factory.cc`.

8.15.3.3 `PointContactFacetFactory& jeod::PointContactFacetFactory::operator= ( const PointContactFacetFactory & rhs )` `[private]`

### 8.15.4 Friends And Related Function Documentation

8.15.4.1 `void init_attrjeod__PointContactFacetFactory ( )` `[friend]`

8.15.4.2 `friend class InputProcessor` `[friend]`

Definition at line 53 of file `point_contact_facet_factory.hh`.

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

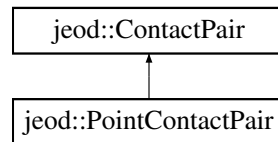
- [point\\_contact\\_facet\\_factory.hh](#)
- [point\\_contact\\_facet\\_factory.cc](#)

## 8.16 jeod::PointContactPair Class Reference

An point to point contact pair for use in the contact model.

```
#include <point_contact_pair.hh>
```

Inheritance diagram for jeod::PointContactPair:



### Public Member Functions

- [PointContactPair](#) ()  
*Default Constructor.*
- virtual [~PointContactPair](#) ()  
*Destructor.*
- virtual void [in\\_contact](#) ()  
*Determine if contact has occurred between the facets of the pair.*
- virtual void [initialize\\_pair](#) ([ContactFacet](#) \*subject\_facet, [ContactFacet](#) \*target\_facet)  
*Initialize the contact pair by setting the subject, target, and creating the relstate if possible.*

### Data Fields

- [PointContactFacet](#) \* [point\\_subject](#)  
*pointer to the point contact facet that is the subject of the associated relative states.*
- [PointContactFacet](#) \* [point\\_target](#)  
*pointer to the point contact facet that is the target of the associated relative states.*

### Private Member Functions

- [PointContactPair](#) & [operator=](#) (const [PointContactPair](#) &rhs)
- [PointContactPair](#) (const [PointContactPair](#) &rhs)

### Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_PointContactPair](#) ()

### Additional Inherited Members

#### 8.16.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 48 of file point\_contact\_pair.hh.

## 8.16.2 Constructor & Destructor Documentation

### 8.16.2.1 jeod::PointContactPair::PointContactPair ( void )

Default Constructor.

Definition at line 47 of file point\_contact\_pair.cc.

### 8.16.2.2 jeod::PointContactPair::~~PointContactPair ( void ) [virtual]

Destructor.

Definition at line 60 of file point\_contact\_pair.cc.

### 8.16.2.3 jeod::PointContactPair::PointContactPair ( const PointContactPair & rhs ) [private]

## 8.16.3 Member Function Documentation

### 8.16.3.1 void jeod::PointContactPair::in\_contact ( void ) [virtual]

Determine if contact has occurred between the facets of the pair.

Implements [jeod::ContactPair](#).

Definition at line 70 of file point\_contact\_pair.cc.

References [jeod::PointContactFacet::calculate\\_contact\\_point\(\)](#), [jeod::PairInteraction::calculate\\_forces\(\)](#), [jeod::PointContactFacet::contact\\_point](#), [jeod::ContactPair::interaction](#), [point\\_subject](#), [point\\_target](#), [jeod::PointContactFacet::radius](#), and [jeod::ContactPair::rel\\_state](#).

### 8.16.3.2 void jeod::PointContactPair::initialize\_pair ( ContactFacet \* subject\_facet, ContactFacet \* target\_facet ) [virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

in, out	<i>subject_facet</i>	subject <a href="#">ContactFacet</a>
in, out	<i>target_facet</i>	target <a href="#">ContactFacet</a>

Implements [jeod::ContactPair](#).

Definition at line 125 of file point\_contact\_pair.cc.

References [jeod::ContactUtils::copy\\_const\\_char\\_to\\_char\(\)](#), [point\\_subject](#), [point\\_target](#), [jeod::ContactPair::rel\\_state](#), [jeod::ContactPair::subject](#), [jeod::ContactPair::target](#), and [jeod::ContactFacet::vehicle\\_point](#).

Referenced by [jeod::PointContactFacet::create\\_pair\(\)](#).

### 8.16.3.3 PointContactPair& jeod::PointContactPair::operator= ( const PointContactPair & rhs ) [private]

## 8.16.4 Friends And Related Function Documentation

### 8.16.4.1 void init\_attrjeod\_\_PointContactPair ( ) [friend]

### 8.16.4.2 friend class InputProcessor [friend]

Definition at line 50 of file point\_contact\_pair.hh.

### 8.16.5 Field Documentation

#### 8.16.5.1 `PointContactFacet*` `jeod::PointContactPair::point_subject`

pointer to the point contact facet that is the subject of the associated relative states.

`trick_units(-)`

Definition at line 56 of file `point_contact_pair.hh`.

Referenced by `in_contact()`, and `initialize_pair()`.

#### 8.16.5.2 `PointContactFacet*` `jeod::PointContactPair::point_target`

pointer to the point contact facet that is the target of the associated relative states.

`trick_units(-)`

Definition at line 61 of file `point_contact_pair.hh`.

Referenced by `in_contact()`, and `initialize_pair()`.

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

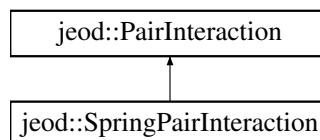
- [point\\_contact\\_pair.hh](#)
- [point\\_contact\\_pair.cc](#)

## 8.17 `jeod::SpringPairInteraction` Class Reference

Simple spring contact parameters.

```
#include <spring_pair_interaction.hh>
```

Inheritance diagram for `jeod::SpringPairInteraction`:



### Public Member Functions

- [SpringPairInteraction](#) ()  
*Default Constructor.*
- virtual [~SpringPairInteraction](#) ()  
*Destructor.*
- virtual void [calculate\\_forces](#) ([ContactFacet](#) \*subject, [ContactFacet](#) \*target, [RelativeDerivedState](#) \*rel\_state, double \*penetration\_vector, double \*rel\_velocity)  
*force calculation for a simple spring based contact dynamics model, takes in geometry information from the appropriate [ContactFacet::calculate\\_forces](#) but doesn't know about specific type of [ContactFacet](#)*

### Data Fields

- double [spring\\_k](#)  
*Spring stiffness constant.*
- double [damping\\_b](#)

*Spring damping constant.*

- double [mu](#)

*Coefficient of friction.*

## Private Member Functions

- [SpringPairInteraction](#) & [operator=](#) (const [SpringPairInteraction](#) &rhs)
- [SpringPairInteraction](#) (const [SpringPairInteraction](#) &rhs)

## Friends

- class [InputProcessor](#)
- void [init\\_attrjeod\\_\\_SpringPairInteraction](#) ()

## 8.17.1 Detailed Description

Simple spring contact parameters.

Definition at line 47 of file `spring_pair_interaction.hh`.

## 8.17.2 Constructor & Destructor Documentation

### 8.17.2.1 jeod::SpringPairInteraction::SpringPairInteraction ( void )

Default Constructor.

Definition at line 51 of file `spring_pair_interaction.cc`.

### 8.17.2.2 jeod::SpringPairInteraction::~~SpringPairInteraction ( void ) [virtual]

Destructor.

Definition at line 66 of file `spring_pair_interaction.cc`.

### 8.17.2.3 jeod::SpringPairInteraction::SpringPairInteraction ( const SpringPairInteraction & rhs ) [private]

## 8.17.3 Member Function Documentation

### 8.17.3.1 void jeod::SpringPairInteraction::calculate\_forces ( ContactFacet \* subject, ContactFacet \* target, RelativeDerivedState \* rel\_state, double \* penetration\_vector, double \* rel\_velocity ) [virtual]

force calculation for a simple spring based contact dynamics model, takes in geometry information from the appropriate `ContactFacet::calculate_forces` but doesn't know about specific type of [ContactFacet](#)

Parameters

in, out	<i>subject</i>	subject frame of the relative state
in, out	<i>target</i>	target frame of the relative state
in	<i>rel_state</i>	relative state between subject and target in subject frame
in	<i>penetration_vector</i>	vector that characterises the interpenetration of the subject and the target

<code>in</code>	<code>rel_velocity</code>	relative velocity of the subject and the target in the subject frame
-----------------	---------------------------	--

Implements [jeod::PairInteraction](#).

Definition at line 87 of file `spring_pair_interaction.cc`.

References `jeod::ContactFacet::calculate_torque()`, `damping_b`, `jeod::PairInteraction::friction_mag`, `mu`, `spring_k`, `jeod::ContactFacet::vehicle_body`, and `jeod::ContactFacet::vehicle_point`.

**8.17.3.2 SpringPairInteraction& jeod::SpringPairInteraction::operator= ( const SpringPairInteraction & rhs )**  
[private]

## 8.17.4 Friends And Related Function Documentation

**8.17.4.1 void init\_attrjeod\_SpringPairInteraction ( )** [friend]

**8.17.4.2 friend class InputProcessor** [friend]

Definition at line 50 of file `spring_pair_interaction.hh`.

## 8.17.5 Field Documentation

**8.17.5.1 double jeod::SpringPairInteraction::damping\_b**

Spring damping constant.

trick\_units(N\*s/m)

Definition at line 61 of file `spring_pair_interaction.hh`.

Referenced by `calculate_forces()`.

**8.17.5.2 double jeod::SpringPairInteraction::mu**

Coefficient of friction.

trick\_units(-)

Definition at line 66 of file `spring_pair_interaction.hh`.

Referenced by `calculate_forces()`.

**8.17.5.3 double jeod::SpringPairInteraction::spring\_k**

Spring stiffness constant.

trick\_units(N/m)

Definition at line 56 of file `spring_pair_interaction.hh`.

Referenced by `calculate_forces()`.

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

- [spring\\_pair\\_interaction.hh](#)
- [spring\\_pair\\_interaction.cc](#)



## Chapter 9

# File Documentation

### 9.1 `class_declarations.hh` File Reference

Forward declaration of classes defined in the contact model.

#### Namespaces

- [jeod](#)  
*Namespace jeod.*

#### 9.1.1 Detailed Description

Forward declaration of classes defined in the contact model.

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

### 9.2 `contact.cc` File Reference

Base Contact for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "../include/contact.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_pair.hh"
#include "../include/pair_interaction.hh"
```

#### Namespaces

- [jeod](#)  
*Namespace jeod.*

#### 9.2.1 Detailed Description

Base Contact for use with contact interaction model.

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

### 9.3 contact.hh File Reference

(Base class to for the contact manager for use with contact interaction model)

```
#include <list>
#include "dynamics/dyn_manager/include/class_declarations.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/container/include/pointer_list.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
#include "contact_pair.hh"
#include "pair_interaction.hh"
```

#### Data Structures

- class [jeod::Contact](#)

*An base contact class for use in the surface model.*

#### Namespaces

- [jeod](#)

*Namespace jeod.*

#### 9.3.1 Detailed Description

(Base class to for the contact manager for use with contact interaction model)

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

### 9.4 contact\_facet.cc File Reference

Define ContactFacet::create\_vehicle\_point.

```
#include <cstdint>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/mass/include/mass_point_init.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_utils.hh"
```

#### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.4.1 Detailed Description

Define `ContactFacet::create_vehicle_point`.

Definition in file [contact\\_facet.cc](#).

## 9.5 `contact_facet.hh` File Reference

Individual facets for use with contact interaction models.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "class_declarations.hh"
```

### Data Structures

- class [jeod::ContactFacet](#)

*An contact interaction specific facet for use in the surface model.*

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.5.1 Detailed Description

Individual facets for use with contact interaction models.

Definition in file [contact\\_facet.hh](#).

## 9.6 `contact_messages.cc` File Reference

Implement `contact_messages`.

```
#include "../include/contact_messages.hh"
```

### Namespaces

- [jeod](#)

*Namespace jeod.*

### Macros

- `#define` [PATH](#) "interactions/contact"

### 9.6.1 Detailed Description

Implement `contact_messages`.

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

## 9.7 `contact_messages.hh` File Reference

Contact message for message handling.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/message/include/message_handler.hh"
```

### Data Structures

- class [jeod::ContactMessages](#)  
*Messages associated with use of the contact model.*

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.7.1 Detailed Description

Contact message for message handling.

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

## 9.8 `contact_pair.cc` File Reference

ContactPair class for use with contact interaction model.

```
#include "utils/math/include/vector3.hh"
#include "dynamics/mass/include/mass.hh"
#include "../include/contact_pair.hh"
```

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.8.1 Detailed Description

ContactPair class for use with contact interaction model.

Definition in file [contact\\_pair.cc](#).

## 9.9 contact\_pair.hh File Reference

Base class for pair of contact facets for use with contact interaction model.

```
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
```

### Data Structures

- class [jeod::ContactPair](#)

*An base contact pair class for use in the contact model.*

### Namespaces

- [jeod](#)

*Namespace jeod.*

#### 9.9.1 Detailed Description

Base class for pair of contact facets for use with contact interaction model.

Definition in file [contact\\_pair.hh](#).

## 9.10 contact\_params.cc File Reference

contact parameters for use in the surface model

```
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_params.hh"
```

### Namespaces

- [jeod](#)

*Namespace jeod.*

#### 9.10.1 Detailed Description

contact parameters for use in the surface model

Definition in file [contact\\_params.cc](#).

## 9.11 contact\_params.hh File Reference

A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/facet_params.hh"
```

## Data Structures

- class [jeod::ContactParams](#)

*A base class for all contact parameters used in the surface model.*

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.11.1 Detailed Description

A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.

Definition in file [contact\\_params.hh](#).

## 9.12 contact\_surface.cc File Reference

Vehicle surface model for the contact interaction models.

```
#include <typeinfo>
#include <cstdint>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "../include/contact_surface.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_facet.hh"
```

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.12.1 Detailed Description

Vehicle surface model for the contact interaction models.

Definition in file [contact\\_surface.cc](#).

## 9.13 contact\_surface.hh File Reference

Vehicle surface model for contact.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "contact_facet.hh"
```

## Data Structures

- class [jeod::ContactSurface](#)

*The contact specific interaction surface, for use with the surface model.*

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.13.1 Detailed Description

Vehicle surface model for contact.

Definition in file [contact\\_surface.hh](#).

## 9.14 contact\_surface\_factory.cc File Reference

Factory that creates an contact surface, from a surface model.

```
#include <cstring>
#include <cstdint>
#include "dynamics/mass/include/mass.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/facet_params.hh"
#include "utils/surface_model/include/surface_model.hh"
#include "utils/surface_model/include/interaction_surface.hh"
#include "../include/contact_surface_factory.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.14.1 Detailed Description

Factory that creates an contact surface, from a surface model.

Definition in file [contact\\_surface\\_factory.cc](#).

## 9.15 contact\_surface\_factory.hh File Reference

Factory that creates an contact interaction surface from a surface model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface_factory.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "point_contact_facet_factory.hh"
#include "line_contact_facet_factory.hh"
```

## Data Structures

- class [jeod::ContactSurfaceFactory](#)

*The surface factory that creates an contact specific surface from a general surface.*

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.15.1 Detailed Description

Factory that creates an contact interaction surface from a surface model.

Definition in file [contact\\_surface\\_factory.hh](#).

## 9.16 contact\_utils.hh File Reference

This Model is used for utility routines.

```
#include "contact_utils_inline.hh"
```

## Data Structures

- class [jeod::ContactUtils](#)

*Utility string and math functions for the contact model.*

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.16.1 Detailed Description

This Model is used for utility routines.

Definition in file [contact\\_utils.hh](#).

## 9.17 contact\_utils\_inline.hh File Reference

Define [ContactUtils::create\\_relstate\\_name](#), [ContactUtils::copy\\_const\\_char\\_to\\_char](#).

```
#include <cstring>
#include "utils/math/include/vector3.hh"
#include "contact_utils.hh"
#include "contact_messages.hh"
```



## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.17.1 Detailed Description

Define ContactUtils::create\_relstate\_name, ContactUtils::copy\_const\_char\_to\_char.

Definition in file [contact\\_utils\\_inline.hh](#).

## 9.18 line\_contact\_facet.cc File Reference

Define LineContactFacet functions.

```
#include <cstring>
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/numerical.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
#include "../include/line_contact_facet.hh"
#include "../include/line_contact_pair.hh"
#include "../include/line_point_contact_pair.hh"
#include "../include/contact_utils.hh"
#include "../include/contact.hh"
```

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.18.1 Detailed Description

Define LineContactFacet functions.

Definition in file [line\\_contact\\_facet.cc](#).

## 9.19 line\_contact\_facet.hh File Reference

The contact facet based on the distance to a line segment centered on the vehicle point.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "class_declarations.hh"
#include "point_contact_facet.hh"
#include "line_contact_pair.hh"
#include "line_point_contact_pair.hh"
```

## Data Structures

- class [jeod::LineContactFacet](#)

*The contact facet based on the distance to a single point, specifically the vehicle point.*

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.19.1 Detailed Description

The contact facet based on the distance to a line segment centered on the vehicle point. In effect this represents a cylinder with spherical ends.

Definition in file [line\\_contact\\_facet.hh](#).

## 9.20 line\_contact\_facet\_factory.cc File Reference

Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.

```
#include <typeinfo>
#include <cstdlib>
#include "utils/surface_model/include/cylinder.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/line_contact_facet_factory.hh"
#include "../include/line_contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.20.1 Detailed Description

Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.

Definition in file [line\\_contact\\_facet\\_factory.cc](#).

## 9.21 line\_contact\_facet\_factory.hh File Reference

Creates a line contact facet from an cylinder facet.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "class_declarations.hh"
#include "line_contact_facet.hh"
```

## Data Structures

- class [jeod::LineContactFacetFactory](#)  
*Creates a [PointContactFacet](#) from an [InteractionFacet](#).*

## Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.21.1 Detailed Description

Creates a line contact facet from an cylinder facet.

Definition in file [line\\_contact\\_facet\\_factory.hh](#).

## 9.22 line\_contact\_pair.cc File Reference

LineContactPair class for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/line_contact_pair.hh"
#include "../include/line_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

## Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.22.1 Detailed Description

LineContactPair class for use with contact interaction model.

Definition in file [line\\_contact\\_pair.cc](#).

## 9.23 line\_contact\_pair.hh File Reference

Class for a pair of line contact facets for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

## Data Structures

- class [jeod::LineContactPair](#)  
*An point to point contact pair for use in the contact model.*

## Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.23.1 Detailed Description

Class for a pair of line contact facets for use with contact interaction model.

Definition in file [line\\_contact\\_pair.hh](#).

## 9.24 line\_point\_contact\_pair.cc File Reference

LinePointContactPair class for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/line_point_contact_pair.hh"
#include "../include/line_contact_facet.hh"
#include "../include/point_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

## Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.24.1 Detailed Description

LinePointContactPair class for use with contact interaction model.

Definition in file [line\\_point\\_contact\\_pair.cc](#).

## 9.25 line\_point\_contact\_pair.hh File Reference

Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

## Data Structures

- class [jeod::LinePointContactPair](#)  
*An point to point contact pair for use in the contact model.*

## Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.25.1 Detailed Description

Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.

Definition in file [line\\_point\\_contact\\_pair.hh](#).

## 9.26 pair\_interaction.cc File Reference

A class to define the interaction type for a pair of contact facets.

```
#include <cstring>
#include "../include/pair_interaction.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_params.hh"
```

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.26.1 Detailed Description

A class to define the interaction type for a pair of contact facets. This is a base class and derived classes define the force generation function when contact between facets occurs.

Definition in file [pair\\_interaction.cc](#).

## 9.27 pair\_interaction.hh File Reference

A class to define the interaction type for a pair of contact facets.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/class_declarations.hh"
#include "../include/class_declarations.hh"
```

### Data Structures

- class [jeod::PairInteraction](#)  
*Simple spring contact parameters.*

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.27.1 Detailed Description

A class to define the interaction type for a pair of contact facets. This is a base class and derived classes define the force generation function when contact between facets occurs.

Definition in file [pair\\_interaction.hh](#).

## 9.28 point\_contact\_facet.cc File Reference

Define PointContactFacet functions.

```
#include <cstring>
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/numerical.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
#include "../include/point_contact_facet.hh"
#include "../include/point_contact_pair.hh"
#include "../include/contact_utils.hh"
#include "../include/contact.hh"
```

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.28.1 Detailed Description

Define PointContactFacet functions.

Definition in file [point\\_contact\\_facet.cc](#).

## 9.29 point\_contact\_facet.hh File Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
#include "point_contact_pair.hh"
```

### Data Structures

- class [jeod::PointContactFacet](#)

*The contact facet based on the distance to a single point, specifically the vehicle point.*

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.29.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point. In effect this represents a sphere.

Definition in file [point\\_contact\\_facet.hh](#).

## 9.30 point\_contact\_facet\_factory.cc File Reference

Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.

```
#include <typeinfo>
#include <cstdint>
#include "utils/surface_model/include/flat_plate_circular.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/point_contact_facet_factory.hh"
#include "../include/point_contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.30.1 Detailed Description

Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.

Definition in file [point\\_contact\\_facet\\_factory.cc](#).

## 9.31 point\_contact\_facet\_factory.hh File Reference

Creates a point contact facet from an circular flat plate facet.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "class_declarations.hh"
#include "point_contact_facet.hh"
```

### Data Structures

- class [jeod::PointContactFacetFactory](#)

*Creates a [PointContactFacet](#) from an [InteractionFacet](#).*

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.31.1 Detailed Description

Creates a point contact facet from an circular flat plate facet.

Definition in file [point\\_contact\\_facet\\_factory.hh](#).

## 9.32 point\_contact\_pair.cc File Reference

ContactPair class for use with contact interaction model.

```
#include "utils/named_item/include/named_item.hh"
#include "../include/point_contact_pair.hh"
#include "../include/point_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

### Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.32.1 Detailed Description

ContactPair class for use with contact interaction model.

Definition in file [point\\_contact\\_pair.cc](#).

## 9.33 point\_contact\_pair.hh File Reference

Class for a pair of point contact facets for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

### Data Structures

- class [jeod::PointContactPair](#)

*An point to point contact pair for use in the contact model.*

### Namespaces

- [jeod](#)

*Namespace jeod.*



### 9.33.1 Detailed Description

Class for a pair of point contact facets for use with contact interaction model.

Definition in file [point\\_contact\\_pair.hh](#).

## 9.34 `spring_pair_interaction.cc` File Reference

spring pair interaction for use in the contact model

```
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "dynamics/dyn_body/include/body_ref_frame.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "../include/spring_pair_interaction.hh"
#include "../include/contact_facet.hh"
```

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.34.1 Detailed Description

spring pair interaction for use in the contact model

Definition in file [spring\\_pair\\_interaction.cc](#).

## 9.35 `spring_pair_interaction.hh` File Reference

A class for pair interactions based on a simple spring.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "pair_interaction.hh"
```

### Data Structures

- class [jeod::SpringPairInteraction](#)  
*Simple spring contact parameters.*

### Namespaces

- [jeod](#)  
*Namespace jeod.*

### 9.35.1 Detailed Description

A class for pair interactions based on a simple spring.

Definition in file [spring\\_pair\\_interaction.hh](#).

# Index

- ~Contact
  - jeod::Contact, [18](#)
- ~ContactFacet
  - jeod::ContactFacet, [24](#)
- ~ContactPair
  - jeod::ContactPair, [31](#)
- ~ContactParams
  - jeod::ContactParams, [35](#)
- ~ContactSurface
  - jeod::ContactSurface, [37](#)
- ~ContactSurfaceFactory
  - jeod::ContactSurfaceFactory, [39](#)
- ~LineContactFacet
  - jeod::LineContactFacet, [44](#)
- ~LineContactFacetFactory
  - jeod::LineContactFacetFactory, [47](#)
- ~LineContactPair
  - jeod::LineContactPair, [49](#)
- ~LinePointContactPair
  - jeod::LinePointContactPair, [51](#)
- ~PairInteraction
  - jeod::PairInteraction, [54](#)
- ~PointContactFacet
  - jeod::PointContactFacet, [57](#)
- ~PointContactFacetFactory
  - jeod::PointContactFacetFactory, [60](#)
- ~PointContactPair
  - jeod::PointContactPair, [63](#)
- ~SpringPairInteraction
  - jeod::SpringPairInteraction, [65](#)
- active
  - jeod::Contact, [22](#)
  - jeod::ContactFacet, [25](#)
- add\_facet\_params
  - jeod::ContactSurfaceFactory, [40](#)
- allocate\_array
  - jeod::ContactSurface, [37](#)
- allocate\_interaction\_facet
  - jeod::ContactSurface, [37](#)
- calculate\_contact\_point
  - jeod::LineContactFacet, [44](#)
  - jeod::PointContactFacet, [57](#)
- calculate\_forces
  - jeod::PairInteraction, [54](#)
  - jeod::SpringPairInteraction, [65](#)
- calculate\_torque
  - jeod::ContactFacet, [24](#)
  - jeod::LineContactFacet, [44](#)
  - jeod::PointContactFacet, [57](#)
- check\_contact
  - jeod::Contact, [18](#)
- check\_tree
  - jeod::ContactPair, [31](#)
- class\_declarations.hh, [67](#)
- collect\_forces\_torques
  - jeod::ContactSurface, [37](#)
- Contact, [13](#)
  - jeod::Contact, [18](#)
  - PATH, [14](#)
- contact.cc, [67](#)
- contact.hh, [68](#)
- contact\_facet.cc, [68](#)
- contact\_facet.hh, [69](#)
- contact\_facets
  - jeod::ContactSurface, [38](#)
- contact\_force
  - jeod::ContactSurface, [38](#)
- contact\_limit\_factor
  - jeod::Contact, [22](#)
- contact\_messages.cc, [69](#)
- contact\_messages.hh, [70](#)
- contact\_pair.cc, [70](#)
- contact\_pair.hh, [71](#)
- contact\_pairs
  - jeod::Contact, [22](#)
- contact\_params.cc, [71](#)
- contact\_params.hh, [71](#)
- contact\_point
  - jeod::PointContactFacet, [59](#)
- contact\_surface.cc, [72](#)
- contact\_surface.hh, [72](#)
- contact\_surface\_factory.cc, [73](#)
- contact\_surface\_factory.hh, [73](#)
- contact\_torque
  - jeod::ContactSurface, [38](#)
- contact\_utils.hh, [74](#)
- contact\_utils\_inline.hh, [74](#)
- ContactFacet
  - jeod::ContactFacet, [24](#)
- ContactMessages
  - jeod::ContactMessages, [28](#)
- ContactPair
  - jeod::ContactPair, [31](#)
- ContactParams
  - jeod::ContactParams, [35](#)
- ContactSurface
  - jeod::ContactSurface, [37](#)

- ContactSurfaceFactory
  - jeod::ContactSurfaceFactory, [39](#), [40](#)
- copy\_const\_char\_to\_char
  - jeod::ContactUtils, [41](#)
- create\_facet
  - jeod::LineContactFacetFactory, [47](#)
  - jeod::PointContactFacetFactory, [61](#)
- create\_pair
  - jeod::ContactFacet, [24](#), [25](#)
  - jeod::LineContactFacet, [44](#), [45](#)
  - jeod::PointContactFacet, [58](#)
- create\_relstate\_name
  - jeod::ContactUtils, [42](#)
- create\_surface
  - jeod::ContactSurfaceFactory, [40](#)
- create\_vehicle\_point
  - jeod::ContactFacet, [25](#)
- damping\_b
  - jeod::SpringPairInteraction, [66](#)
- dist\_line\_segments
  - jeod::ContactUtils, [42](#)
- dyn\_manager
  - jeod::Contact, [22](#)
- facets\_size
  - jeod::ContactSurface, [38](#)
- find\_interaction
  - jeod::Contact, [18](#)
- friction\_mag
  - jeod::PairInteraction, [55](#)
- get\_name
  - jeod::ContactFacet, [25](#)
- get\_subject
  - jeod::ContactPair, [31](#)
- get\_target
  - jeod::ContactPair, [31](#)
- in\_contact
  - jeod::ContactPair, [32](#)
  - jeod::LineContactPair, [49](#)
  - jeod::LinePointContactPair, [52](#)
  - jeod::PointContactPair, [63](#)
- in\_range
  - jeod::ContactPair, [32](#)
- init\_attrjeod\_\_Contact
  - jeod::Contact, [21](#)
- init\_attrjeod\_\_ContactFacet
  - jeod::ContactFacet, [25](#)
- init\_attrjeod\_\_ContactMessages
  - jeod::ContactMessages, [28](#)
- init\_attrjeod\_\_ContactPair
  - jeod::ContactPair, [33](#)
- init\_attrjeod\_\_ContactParams
  - jeod::ContactParams, [35](#)
- init\_attrjeod\_\_ContactSurface
  - jeod::ContactSurface, [38](#)
- init\_attrjeod\_\_ContactSurfaceFactory
  - jeod::ContactSurfaceFactory, [40](#)
- init\_attrjeod\_\_LineContactFacet
  - jeod::LineContactFacet, [45](#)
- init\_attrjeod\_\_LineContactFacetFactory
  - jeod::LineContactFacetFactory, [48](#)
- init\_attrjeod\_\_LineContactPair
  - jeod::LineContactPair, [50](#)
- init\_attrjeod\_\_LinePointContactPair
  - jeod::LinePointContactPair, [52](#)
- init\_attrjeod\_\_PairInteraction
  - jeod::PairInteraction, [55](#)
- init\_attrjeod\_\_PointContactFacet
  - jeod::PointContactFacet, [59](#)
- init\_attrjeod\_\_PointContactFacetFactory
  - jeod::PointContactFacetFactory, [61](#)
- init\_attrjeod\_\_PointContactPair
  - jeod::PointContactPair, [63](#)
- init\_attrjeod\_\_SpringPairInteraction
  - jeod::SpringPairInteraction, [66](#)
- initialization\_error
  - jeod::ContactMessages, [28](#)
- initialization\_warns
  - jeod::ContactMessages, [28](#)
- initialize\_contact
  - jeod::Contact, [19](#)
- initialize\_pair
  - jeod::ContactPair, [32](#)
  - jeod::LineContactPair, [49](#)
  - jeod::LinePointContactPair, [52](#)
  - jeod::PointContactPair, [63](#)
- initialize\_relstate
  - jeod::ContactPair, [32](#)
- InputProcessor
  - jeod::Contact, [21](#)
  - jeod::ContactFacet, [25](#)
  - jeod::ContactMessages, [28](#)
  - jeod::ContactPair, [33](#)
  - jeod::ContactParams, [35](#)
  - jeod::ContactSurface, [38](#)
  - jeod::ContactSurfaceFactory, [40](#)
  - jeod::LineContactFacet, [45](#)
  - jeod::LineContactFacetFactory, [48](#)
  - jeod::LineContactPair, [50](#)
  - jeod::LinePointContactPair, [52](#)
  - jeod::PairInteraction, [55](#)
  - jeod::PointContactFacet, [59](#)
  - jeod::PointContactFacetFactory, [61](#)
  - jeod::PointContactPair, [63](#)
  - jeod::SpringPairInteraction, [66](#)
- interaction
  - jeod::ContactPair, [33](#)
- interaction\_distance
  - jeod::ContactPair, [33](#)
- Interactions, [12](#)
- is\_active
  - jeod::ContactPair, [32](#)
- is\_complete
  - jeod::ContactPair, [33](#)

- is\_correct\_factory
  - jeod::LineContactFacetFactory, 47
  - jeod::PointContactFacetFactory, 61
- is\_correct\_interaction
  - jeod::PairInteraction, 54
- jeod, 15
- jeod::Contact, 17
  - ~Contact, 18
  - active, 22
  - check\_contact, 18
  - Contact, 18
  - contact\_limit\_factor, 22
  - contact\_pairs, 22
  - dyn\_manager, 22
  - find\_interaction, 18
  - init\_attrjeod\_\_Contact, 21
  - initialize\_contact, 19
  - InputProcessor, 21
  - operator=, 19
  - pair\_interactions, 22
  - register\_contact, 19, 21
  - register\_interaction, 21
  - unique\_pair, 21
- jeod::ContactFacet, 23
  - ~ContactFacet, 24
  - active, 25
  - calculate\_torque, 24
  - ContactFacet, 24
  - create\_pair, 24, 25
  - create\_vehicle\_point, 25
  - get\_name, 25
  - init\_attrjeod\_\_ContactFacet, 25
  - InputProcessor, 25
  - max\_dimension, 26
  - normal, 26
  - operator=, 25
  - position, 26
  - set\_max\_dimension, 25
  - surface\_type, 26
  - vehicle\_body, 26
  - vehicle\_point, 26
- jeod::ContactMessages, 27
  - ContactMessages, 28
  - init\_attrjeod\_\_ContactMessages, 28
  - initialization\_error, 28
  - initialization\_warns, 28
  - InputProcessor, 28
  - operator=, 28
  - pre\_initialization\_error, 28
  - runtime\_error, 29
  - runtime\_inform, 29
  - runtime\_warns, 29
- jeod::ContactPair, 29
  - ~ContactPair, 31
  - check\_tree, 31
  - ContactPair, 31
  - get\_subject, 31
  - get\_target, 31
  - in\_contact, 32
  - in\_range, 32
  - init\_attrjeod\_\_ContactPair, 33
  - initialize\_pair, 32
  - initialize\_relstate, 32
  - InputProcessor, 33
  - interaction, 33
  - interaction\_distance, 33
  - is\_active, 32
  - is\_complete, 33
  - operator=, 33
  - rel\_state, 33
  - subject, 34
  - target, 34
- jeod::ContactParams, 34
  - ~ContactParams, 35
  - ContactParams, 35
  - init\_attrjeod\_\_ContactParams, 35
  - InputProcessor, 35
  - operator=, 35
- jeod::ContactSurface, 35
  - ~ContactSurface, 37
  - allocate\_array, 37
  - allocate\_interaction\_facet, 37
  - collect\_forces\_torques, 37
  - contact\_facets, 38
  - contact\_force, 38
  - contact\_torque, 38
  - ContactSurface, 37
  - facets\_size, 38
  - init\_attrjeod\_\_ContactSurface, 38
  - InputProcessor, 38
  - operator=, 37
- jeod::ContactSurfaceFactory, 38
  - ~ContactSurfaceFactory, 39
  - add\_facet\_params, 40
  - ContactSurfaceFactory, 39, 40
  - create\_surface, 40
  - init\_attrjeod\_\_ContactSurfaceFactory, 40
  - InputProcessor, 40
  - line\_contact\_facet\_factory, 40
  - operator=, 40
  - point\_contact\_facet\_factory, 41
- jeod::ContactUtils, 41
  - copy\_const\_char\_to\_char, 41
  - create\_relstate\_name, 42
  - dist\_line\_segments, 42
- jeod::LineContactFacet, 43
  - ~LineContactFacet, 44
  - calculate\_contact\_point, 44
  - calculate\_torque, 44
  - create\_pair, 44, 45
  - init\_attrjeod\_\_LineContactFacet, 45
  - InputProcessor, 45
  - length, 46
  - LineContactFacet, 44
  - operator=, 45
  - set\_max\_dimension, 45

- jeod::LineContactFacetFactory, 46
  - ~LineContactFacetFactory, 47
  - create\_facet, 47
  - init\_attrjeod\_\_LineContactFacetFactory, 48
  - InputProcessor, 48
  - is\_correct\_factory, 47
  - LineContactFacetFactory, 47
  - operator=, 48
- jeod::LineContactPair, 48
  - ~LineContactPair, 49
  - in\_contact, 49
  - init\_attrjeod\_\_LineContactPair, 50
  - initialize\_pair, 49
  - InputProcessor, 50
  - line\_subject, 50
  - line\_target, 50
  - LineContactPair, 49
  - operator=, 50
- jeod::LinePointContactPair, 50
  - ~LinePointContactPair, 51
  - in\_contact, 52
  - init\_attrjeod\_\_LinePointContactPair, 52
  - initialize\_pair, 52
  - InputProcessor, 52
  - line\_subject, 52
  - LinePointContactPair, 51, 52
  - operator=, 52
  - point\_target, 53
- jeod::PairInteraction, 53
  - ~PairInteraction, 54
  - calculate\_forces, 54
  - friction\_mag, 55
  - init\_attrjeod\_\_PairInteraction, 55
  - InputProcessor, 55
  - is\_correct\_interaction, 54
  - operator=, 55
  - PairInteraction, 54
  - params\_1, 55
  - params\_2, 55
- jeod::PointContactFacet, 56
  - ~PointContactFacet, 57
  - calculate\_contact\_point, 57
  - calculate\_torque, 57
  - contact\_point, 59
  - create\_pair, 58
  - init\_attrjeod\_\_PointContactFacet, 59
  - InputProcessor, 59
  - operator=, 58
  - PointContactFacet, 57
  - radius, 59
  - set\_max\_dimension, 58
- jeod::PointContactFacetFactory, 59
  - ~PointContactFacetFactory, 60
  - create\_facet, 61
  - init\_attrjeod\_\_PointContactFacetFactory, 61
  - InputProcessor, 61
  - is\_correct\_factory, 61
  - operator=, 61
  - PointContactFacetFactory, 60
- jeod::PointContactPair, 62
  - ~PointContactPair, 63
  - in\_contact, 63
  - init\_attrjeod\_\_PointContactPair, 63
  - initialize\_pair, 63
  - InputProcessor, 63
  - operator=, 63
  - point\_subject, 64
  - point\_target, 64
  - PointContactPair, 63
- jeod::SpringPairInteraction, 64
  - ~SpringPairInteraction, 65
  - calculate\_forces, 65
  - damping\_b, 66
  - init\_attrjeod\_\_SpringPairInteraction, 66
  - InputProcessor, 66
  - mu, 66
  - operator=, 66
  - spring\_k, 66
  - SpringPairInteraction, 65
- length
  - jeod::LineContactFacet, 46
- line\_contact\_facet.cc, 75
- line\_contact\_facet.hh, 75
- line\_contact\_facet\_factory
  - jeod::ContactSurfaceFactory, 40
- line\_contact\_facet\_factory.cc, 76
- line\_contact\_facet\_factory.hh, 76
- line\_contact\_pair.cc, 77
- line\_contact\_pair.hh, 77
- line\_point\_contact\_pair.cc, 78
- line\_point\_contact\_pair.hh, 78
- line\_subject
  - jeod::LineContactPair, 50
  - jeod::LinePointContactPair, 52
- line\_target
  - jeod::LineContactPair, 50
- LineContactFacet
  - jeod::LineContactFacet, 44
- LineContactFacetFactory
  - jeod::LineContactFacetFactory, 47
- LineContactPair
  - jeod::LineContactPair, 49
- LinePointContactPair
  - jeod::LinePointContactPair, 51, 52
- max\_dimension
  - jeod::ContactFacet, 26
- Models, 11
- mu
  - jeod::SpringPairInteraction, 66
- normal
  - jeod::ContactFacet, 26
- operator=
  - jeod::Contact, 19

- jeod::ContactFacet, [25](#)
- jeod::ContactMessages, [28](#)
- jeod::ContactPair, [33](#)
- jeod::ContactParams, [35](#)
- jeod::ContactSurface, [37](#)
- jeod::ContactSurfaceFactory, [40](#)
- jeod::LineContactFacet, [45](#)
- jeod::LineContactFacetFactory, [48](#)
- jeod::LineContactPair, [50](#)
- jeod::LinePointContactPair, [52](#)
- jeod::PairInteraction, [55](#)
- jeod::PointContactFacet, [58](#)
- jeod::PointContactFacetFactory, [61](#)
- jeod::PointContactPair, [63](#)
- jeod::SpringPairInteraction, [66](#)

PATH

- Contact, [14](#)

pair\_interaction.cc, [79](#)

pair\_interaction.hh, [79](#)

pair\_interactions

- jeod::Contact, [22](#)

PairInteraction

- jeod::PairInteraction, [54](#)

params\_1

- jeod::PairInteraction, [55](#)

params\_2

- jeod::PairInteraction, [55](#)

point\_contact\_facet.cc, [80](#)

point\_contact\_facet.hh, [80](#)

point\_contact\_facet\_factory

- jeod::ContactSurfaceFactory, [41](#)

point\_contact\_facet\_factory.cc, [81](#)

point\_contact\_facet\_factory.hh, [81](#)

point\_contact\_pair.cc, [82](#)

point\_contact\_pair.hh, [82](#)

point\_subject

- jeod::PointContactPair, [64](#)

point\_target

- jeod::LinePointContactPair, [53](#)
- jeod::PointContactPair, [64](#)

PointContactFacet

- jeod::PointContactFacet, [57](#)

PointContactFacetFactory

- jeod::PointContactFacetFactory, [60](#)

PointContactPair

- jeod::PointContactPair, [63](#)

position

- jeod::ContactFacet, [26](#)

pre\_initialization\_error

- jeod::ContactMessages, [28](#)

radius

- jeod::PointContactFacet, [59](#)

register\_contact

- jeod::Contact, [19](#), [21](#)

register\_interaction

- jeod::Contact, [21](#)

rel\_state

- jeod::ContactPair, [33](#)

runtime\_error

- jeod::ContactMessages, [29](#)

runtime\_inform

- jeod::ContactMessages, [29](#)

runtime\_warns

- jeod::ContactMessages, [29](#)

set\_max\_dimension

- jeod::ContactFacet, [25](#)
- jeod::LineContactFacet, [45](#)
- jeod::PointContactFacet, [58](#)

spring\_k

- jeod::SpringPairInteraction, [66](#)

spring\_pair\_interaction.cc, [83](#)

spring\_pair\_interaction.hh, [83](#)

SpringPairInteraction

- jeod::SpringPairInteraction, [65](#)

subject

- jeod::ContactPair, [34](#)

surface\_type

- jeod::ContactFacet, [26](#)

target

- jeod::ContactPair, [34](#)

unique\_pair

- jeod::Contact, [21](#)

vehicle\_body

- jeod::ContactFacet, [26](#)

vehicle\_point

- jeod::ContactFacet, [26](#)