

ContainerModel

5.3

Generated by Doxygen 1.8.14

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Namespace List	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Data Structure Index	7
4.1	Data Structures	7
5	File Index	9
5.1	File List	9
6	Module Documentation	11
6.1	Models	11
6.1.1	Detailed Description	11
6.2	Utils	12
6.2.1	Detailed Description	12
6.3	Container	13
6.3.1	Detailed Description	15
6.3.2	Macro Definition Documentation	15
6.3.2.1	__USE_ISOC99	15
6.3.3	Function Documentation	15
6.3.3.1	operator"!=() [1/3]	15

6.3.3.2	operator"!=() [2 / 3]	16
6.3.3.3	operator"!=() [3 / 3]	16
6.3.3.4	operator<() [1 / 3]	17
6.3.3.5	operator<() [2 / 3]	17
6.3.3.6	operator<() [3 / 3]	18
6.3.3.7	operator<=() [1 / 3]	18
6.3.3.8	operator<=() [2 / 3]	18
6.3.3.9	operator<=() [3 / 3]	19
6.3.3.10	operator==() [1 / 3]	19
6.3.3.11	operator==() [2 / 3]	20
6.3.3.12	operator==() [3 / 3]	20
6.3.3.13	operator>() [1 / 3]	21
6.3.3.14	operator>() [2 / 3]	21
6.3.3.15	operator>() [3 / 3]	21
6.3.3.16	operator>=() [1 / 3]	22
6.3.3.17	operator>=() [2 / 3]	22
6.3.3.18	operator>=() [3 / 3]	23
7	Namespace Documentation	25
7.1	jeod Namespace Reference	25
7.1.1	Detailed Description	26

8 Data Structure Documentation	27
8.1 <code>jeod::JeodAssociativeContainer< ElemType, ContainerType ></code> Class Template Reference	27
8.1.1 Detailed Description	29
8.1.2 Member Typedef Documentation	29
8.1.2.1 <code>base_container_type</code>	29
8.1.2.2 <code>key_compare</code>	29
8.1.2.3 <code>key_type</code>	30
8.1.2.4 <code>this_container_type</code>	30
8.1.2.5 <code>value_compare</code>	30
8.1.3 Constructor & Destructor Documentation	30
8.1.3.1 <code>~JeodAssociativeContainer()</code>	30
8.1.3.2 <code>JeodAssociativeContainer()</code> [1/3]	31
8.1.3.3 <code>JeodAssociativeContainer()</code> [2/3]	31
8.1.3.4 <code>JeodAssociativeContainer()</code> [3/3]	31
8.1.4 Member Function Documentation	31
8.1.4.1 <code>count()</code>	32
8.1.4.2 <code>equal_range()</code> [1/2]	32
8.1.4.3 <code>equal_range()</code> [2/2]	32
8.1.4.4 <code>erase()</code> [1/3]	32
8.1.4.5 <code>erase()</code> [2/3]	33
8.1.4.6 <code>erase()</code> [3/3]	33
8.1.4.7 <code>find()</code> [1/2]	33
8.1.4.8 <code>find()</code> [2/2]	34
8.1.4.9 <code>insert()</code> [1/3]	34
8.1.4.10 <code>insert()</code> [2/3]	34
8.1.4.11 <code>insert()</code> [3/3]	35
8.1.4.12 <code>key_comp()</code>	35
8.1.4.13 <code>lower_bound()</code> [1/2]	35
8.1.4.14 <code>lower_bound()</code> [2/2]	35
8.1.4.15 <code>upper_bound()</code> [1/2]	36

8.1.4.16	upper_bound() [2/2]	36
8.1.4.17	value_comp()	36
8.2	jeod::JeodCheckpointable Class Reference	36
8.2.1	Detailed Description	38
8.2.2	Constructor & Destructor Documentation	38
8.2.2.1	JeodCheckpointable() [1/2]	38
8.2.2.2	~JeodCheckpointable()	38
8.2.2.3	JeodCheckpointable() [2/2]	38
8.2.3	Member Function Documentation	38
8.2.3.1	advance_checkpoint()	38
8.2.3.2	get_final_name()	39
8.2.3.3	get_final_value()	39
8.2.3.4	get_init_name()	39
8.2.3.5	get_init_value()	39
8.2.3.6	get_item_name()	40
8.2.3.7	get_item_value()	40
8.2.3.8	initialize_checkpointable()	40
8.2.3.9	is_checkpoint_finished()	41
8.2.3.10	operator=()	41
8.2.3.11	perform_restore_action()	41
8.2.3.12	post_checkpoint()	41
8.2.3.13	post_restart()	42
8.2.3.14	pre_checkpoint()	42
8.2.3.15	pre_restart()	42
8.2.3.16	start_checkpoint()	43
8.2.3.17	undo_initialize_checkpointable()	43
8.2.4	Friends And Related Function Documentation	43
8.2.4.1	init_attrjeod__JeodCheckpointable	43
8.2.4.2	InputProcessor	43
8.3	jeod::JeodContainer< ContainerType, ElemType > Class Template Reference	44

8.3.1	Detailed Description	45
8.3.2	Member Typedef Documentation	45
8.3.2.1	stl_container_type	45
8.3.2.2	this_container_type	46
8.3.3	Constructor & Destructor Documentation	46
8.3.3.1	JeodContainer() [1/3]	46
8.3.3.2	JeodContainer() [2/3]	46
8.3.3.3	JeodContainer() [3/3]	47
8.3.3.4	~JeodContainer()	47
8.3.4	Member Function Documentation	47
8.3.4.1	advance_checkpoint()	47
8.3.4.2	get_final_name()	48
8.3.4.3	get_init_name()	48
8.3.4.4	get_item_name()	48
8.3.4.5	initialize_checkpointable()	49
8.3.4.6	is_checkpoint_finished()	49
8.3.4.7	operator=() [1/2]	49
8.3.4.8	operator=() [2/2]	50
8.3.4.9	perform_cleanup_action()	50
8.3.4.10	perform_insert_action()	51
8.3.4.11	perform_restore_action()	51
8.3.4.12	start_checkpoint()	52
8.3.4.13	swap_contents() [1/2]	52
8.3.4.14	swap_contents() [2/2]	52
8.3.5	Friends And Related Function Documentation	52
8.3.5.1	init_attrjeod__JeodContainer	52
8.3.5.2	InputProcessor	53
8.3.6	Field Documentation	53
8.3.6.1	checkpoint_iter	53
8.3.6.2	elem_type_descriptor	53

8.4	jeod::JeodList< ElemType > Class Template Reference	54
8.4.1	Detailed Description	55
8.4.2	Member Typedef Documentation	55
8.4.2.1	jeod_sequence_container_type	56
8.4.2.2	jeod_stl_container_type	56
8.4.2.3	stl_container_type	56
8.4.2.4	this_container_type	56
8.4.3	Constructor & Destructor Documentation	56
8.4.3.1	~JeodList()	57
8.4.3.2	JeodList() [1/3]	57
8.4.3.3	JeodList() [2/3]	57
8.4.3.4	JeodList() [3/3]	57
8.4.4	Member Function Documentation	57
8.4.4.1	merge() [1/2]	58
8.4.4.2	merge() [2/2]	58
8.4.4.3	operator=() [1/2]	58
8.4.4.4	operator=() [2/2]	59
8.4.4.5	pop_front()	59
8.4.4.6	push_front()	59
8.4.4.7	remove()	59
8.4.4.8	remove_if()	60
8.4.4.9	reverse()	60
8.4.4.10	sort() [1/2]	60
8.4.4.11	sort() [2/2]	61
8.4.4.12	splice() [1/3]	61
8.4.4.13	splice() [2/3]	61
8.4.4.14	splice() [3/3]	62
8.4.4.15	unique() [1/2]	62
8.4.4.16	unique() [2/2]	62
8.5	jeod::JeodObjectContainer< ContainerType, ElemType > Class Template Reference	63

8.5.1	Detailed Description	64
8.5.2	Constructor & Destructor Documentation	64
8.5.2.1	JeodObjectContainer() [1/3]	64
8.5.2.2	JeodObjectContainer() [2/3]	64
8.5.2.3	JeodObjectContainer() [3/3]	65
8.5.2.4	~JeodObjectContainer()	65
8.5.3	Member Function Documentation	65
8.5.3.1	advance_checkpoint()	65
8.5.3.2	get_final_value()	66
8.5.3.3	get_item_value()	66
8.5.3.4	operator=() [1/2]	66
8.5.3.5	operator=() [2/2]	67
8.5.3.6	perform_cleanup_action()	67
8.5.3.7	perform_insert_action()	67
8.5.3.8	post_checkpoint()	68
8.5.3.9	post_restart()	68
8.5.3.10	pre_checkpoint()	68
8.5.3.11	start_checkpoint()	69
8.5.4	Friends And Related Function Documentation	69
8.5.4.1	init_attrjeod__JeodObjectContainer	69
8.5.4.2	InputProcessor	69
8.5.5	Field Documentation	69
8.5.5.1	copy	69
8.5.5.2	index	70
8.6	jeod::JeodObjectList< ElemType > Class Template Reference	70
8.6.1	Detailed Description	70
8.6.2	Member Typedef Documentation	70
8.6.2.1	type	71
8.7	jeod::JeodObjectSet< ElemType > Class Template Reference	71
8.7.1	Detailed Description	71

8.7.2	Member Typedef Documentation	71
8.7.2.1	type	71
8.8	jeod::JeodObjectVector< ElemType > Class Template Reference	72
8.8.1	Detailed Description	72
8.8.2	Member Typedef Documentation	72
8.8.2.1	type	72
8.9	jeod::JeodPointerContainer< ContainerType, ElemType > Class Template Reference	73
8.9.1	Detailed Description	73
8.9.2	Constructor & Destructor Documentation	74
8.9.2.1	JeodPointerContainer() [1/3]	74
8.9.2.2	JeodPointerContainer() [2/3]	74
8.9.2.3	JeodPointerContainer() [3/3]	74
8.9.2.4	~JeodPointerContainer()	75
8.9.3	Member Function Documentation	75
8.9.3.1	get_item_value()	75
8.9.3.2	initialize_checkpointable()	75
8.9.3.3	operator=() [1/2]	76
8.9.3.4	operator=() [2/2]	76
8.9.3.5	perform_insert_action()	76
8.9.4	Field Documentation	77
8.9.4.1	base_type_descriptor	77
8.10	jeod::JeodPointerList< ElemType > Class Template Reference	77
8.10.1	Detailed Description	77
8.10.2	Member Typedef Documentation	78
8.10.2.1	type	78
8.11	jeod::JeodPointerSet< ElemType > Class Template Reference	78
8.11.1	Detailed Description	78
8.11.2	Member Typedef Documentation	78
8.11.2.1	type	79
8.12	jeod::JeodPointerVector< ElemType > Class Template Reference	79

8.12.1 Detailed Description	79
8.12.2 Member Typedef Documentation	79
8.12.2.1 type	79
8.13 jeod::JeodPrimitiveContainer< ContainerType, ElemType > Class Template Reference	80
8.13.1 Detailed Description	80
8.13.2 Constructor & Destructor Documentation	81
8.13.2.1 JeodPrimitiveContainer() [1/3]	81
8.13.2.2 JeodPrimitiveContainer() [2/3]	81
8.13.2.3 JeodPrimitiveContainer() [3/3]	81
8.13.2.4 ~JeodPrimitiveContainer()	82
8.13.3 Member Function Documentation	82
8.13.3.1 get_item_value()	82
8.13.3.2 operator=() [1/2]	82
8.13.3.3 operator=() [2/2]	83
8.13.3.4 perform_insert_action()	83
8.13.4 Field Documentation	84
8.13.4.1 serializer	84
8.14 jeod::JeodPrimitiveList< ElemType > Class Template Reference	84
8.14.1 Detailed Description	84
8.14.2 Member Typedef Documentation	84
8.14.2.1 type	85
8.15 jeod::JeodPrimitiveSerializer< Type > Class Template Reference	85
8.15.1 Detailed Description	86
8.15.2 Constructor & Destructor Documentation	86
8.15.2.1 JeodPrimitiveSerializer() [1/2]	86
8.15.2.2 ~JeodPrimitiveSerializer()	86
8.15.2.3 JeodPrimitiveSerializer() [2/2]	87
8.15.3 Member Function Documentation	87
8.15.3.1 from_string() [1/5]	87
8.15.3.2 from_string() [2/5]	87

8.15.3.3	from_string() [3/5]	87
8.15.3.4	from_string() [4/5]	88
8.15.3.5	from_string() [5/5]	88
8.15.3.6	operator=()	88
8.15.3.7	to_string() [1/5]	88
8.15.3.8	to_string() [2/5]	89
8.15.3.9	to_string() [3/5]	89
8.15.3.10	to_string() [4/5]	89
8.15.3.11	to_string() [5/5]	89
8.16	jeod::JeodPrimitiveSerializerBase Class Reference	90
8.16.1	Detailed Description	90
8.16.2	Constructor & Destructor Documentation	90
8.16.2.1	JeodPrimitiveSerializerBase()	91
8.16.2.2	~JeodPrimitiveSerializerBase()	91
8.16.3	Member Function Documentation	91
8.16.3.1	deserialize_double()	91
8.16.3.2	deserialize_float()	91
8.16.3.3	deserialize_long_double()	92
8.16.3.4	deserialize_string()	92
8.16.3.5	serialize_double()	93
8.16.3.6	serialize_float()	93
8.16.3.7	serialize_long_double()	93
8.16.3.8	serialize_string()	94
8.17	jeod::JeodPrimitiveSet< ElemType > Class Template Reference	94
8.17.1	Detailed Description	95
8.17.2	Member Typedef Documentation	95
8.17.2.1	type	95
8.18	jeod::JeodPrimitiveVector< ElemType > Class Template Reference	95
8.18.1	Detailed Description	95
8.18.2	Member Typedef Documentation	96

8.18.2.1	type	96
8.19	jeod::JeodSequenceContainer< ElemType, ContainerType > Class Template Reference	96
8.19.1	Detailed Description	98
8.19.2	Member Typedef Documentation	98
8.19.2.1	base_container_type	98
8.19.2.2	this_container_type	98
8.19.3	Constructor & Destructor Documentation	99
8.19.3.1	~JeodSequenceContainer()	99
8.19.3.2	JeodSequenceContainer() [1/3]	99
8.19.3.3	JeodSequenceContainer() [2/3]	99
8.19.3.4	JeodSequenceContainer() [3/3]	99
8.19.4	Member Function Documentation	100
8.19.4.1	assign() [1/2]	100
8.19.4.2	assign() [2/2]	100
8.19.4.3	back() [1/2]	101
8.19.4.4	back() [2/2]	101
8.19.4.5	erase() [1/2]	101
8.19.4.6	erase() [2/2]	101
8.19.4.7	front() [1/2]	102
8.19.4.8	front() [2/2]	102
8.19.4.9	insert() [1/3]	102
8.19.4.10	insert() [2/3]	103
8.19.4.11	insert() [3/3]	103
8.19.4.12	pop_back()	103
8.19.4.13	push_back()	104
8.19.4.14	resize()	104
8.20	jeod::JeodSet< ElemType > Class Template Reference	104
8.20.1	Detailed Description	105
8.20.2	Member Typedef Documentation	106
8.20.2.1	jeod_associative_container_type	106

8.20.2.2	jeod_stl_container_type	106
8.20.2.3	stl_container_type	106
8.20.2.4	this_container_type	106
8.20.3	Constructor & Destructor Documentation	107
8.20.3.1	~JeodSet()	107
8.20.3.2	JeodSet() [1/3]	107
8.20.3.3	JeodSet() [2/3]	107
8.20.3.4	JeodSet() [3/3]	107
8.20.4	Member Function Documentation	108
8.20.4.1	operator=() [1/2]	108
8.20.4.2	operator=() [2/2]	108
8.21	jeod::JeodSTLContainer< ElemType, ContainerType > Class Template Reference	108
8.21.1	Detailed Description	110
8.21.2	Member Typedef Documentation	111
8.21.2.1	allocator_type	111
8.21.2.2	const_iterator	111
8.21.2.3	const_reference	111
8.21.2.4	const_reverse_iterator	112
8.21.2.5	difference_type	112
8.21.2.6	iterator	112
8.21.2.7	reference	112
8.21.2.8	reverse_iterator	113
8.21.2.9	size_type	113
8.21.2.10	this_container_type	113
8.21.2.11	value_type	113
8.21.3	Constructor & Destructor Documentation	113
8.21.3.1	~JeodSTLContainer()	114
8.21.3.2	JeodSTLContainer() [1/3]	114
8.21.3.3	JeodSTLContainer() [2/3]	114
8.21.3.4	JeodSTLContainer() [3/3]	114

8.21.4 Member Function Documentation	115
8.21.4.1 begin() [1/2]	115
8.21.4.2 begin() [2/2]	115
8.21.4.3 clear()	115
8.21.4.4 empty()	115
8.21.4.5 end() [1/2]	116
8.21.4.6 end() [2/2]	116
8.21.4.7 get_allocator()	116
8.21.4.8 insert()	116
8.21.4.9 max_size()	117
8.21.4.10 operator const ContainerType &()	117
8.21.4.11 operator ContainerType &()	117
8.21.4.12 operator=() [1/2]	117
8.21.4.13 operator=() [2/2]	118
8.21.4.14 rbegin() [1/2]	118
8.21.4.15 rbegin() [2/2]	118
8.21.4.16 rend() [1/2]	119
8.21.4.17 rend() [2/2]	119
8.21.4.18 size()	119
8.21.4.19 swap() [1/2]	119
8.21.4.20 swap() [2/2]	120
8.21.5 Field Documentation	120
8.21.5.1 contents	120
8.22 jeod::JeodVector< ElemType > Class Template Reference	121
8.22.1 Detailed Description	122
8.22.2 Member Typedef Documentation	122
8.22.2.1 jeod_sequence_container_type	122
8.22.2.2 jeod_stl_container_type	122
8.22.2.3 stl_container_type	123
8.22.2.4 this_container_type	123

8.22.3	Constructor & Destructor Documentation	123
8.22.3.1	~JeodVector()	123
8.22.3.2	JeodVector() [1/3]	123
8.22.3.3	JeodVector() [2/3]	123
8.22.3.4	JeodVector() [3/3]	123
8.22.4	Member Function Documentation	124
8.22.4.1	at() [1/2]	124
8.22.4.2	at() [2/2]	124
8.22.4.3	capacity()	125
8.22.4.4	operator=() [1/2]	125
8.22.4.5	operator=() [2/2]	125
8.22.4.6	operator[]() [1/2]	125
8.22.4.7	operator[]() [2/2]	126
8.22.4.8	reserve()	126
8.23	jeod::SimpleCheckpointable Class Reference	126
8.23.1	Detailed Description	127
8.23.2	Constructor & Destructor Documentation	127
8.23.2.1	SimpleCheckpointable() [1/2]	128
8.23.2.2	~SimpleCheckpointable()	128
8.23.2.3	SimpleCheckpointable() [2/2]	128
8.23.3	Member Function Documentation	128
8.23.3.1	advance_checkpoint()	128
8.23.3.2	get_init_name()	128
8.23.3.3	get_item_name()	129
8.23.3.4	get_item_value()	129
8.23.3.5	is_checkpoint_finished()	129
8.23.3.6	operator=()	129
8.23.3.7	perform_restore_action()	129
8.23.3.8	simple_restore()	130
8.23.3.9	start_checkpoint()	130
8.23.4	Friends And Related Function Documentation	130
8.23.4.1	init_attrjeod__SimpleCheckpointable	130
8.23.4.2	InputProcessor	130

9	File Documentation	131
9.1	checkpointable.hh File Reference	131
9.1.1	Detailed Description	131
9.2	container.hh File Reference	131
9.2.1	Detailed Description	132
9.3	jeod_associative_container.hh File Reference	132
9.3.1	Detailed Description	132
9.4	jeod_container_compare.hh File Reference	132
9.4.1	Detailed Description	134
9.5	jeod_list.hh File Reference	134
9.5.1	Detailed Description	134
9.6	jeod_sequence_container.hh File Reference	135
9.6.1	Detailed Description	135
9.7	jeod_set.hh File Reference	135
9.7.1	Detailed Description	136
9.8	jeod_stl_container.hh File Reference	136
9.8.1	Detailed Description	136
9.9	jeod_vector.hh File Reference	136
9.9.1	Detailed Description	137
9.10	object_container.hh File Reference	137
9.10.1	Detailed Description	137
9.10.2	Macro Definition Documentation	137
9.10.2.1	JEOD_OBJECT_CONTAINER	138
9.11	object_list.hh File Reference	138
9.11.1	Detailed Description	138
9.12	object_set.hh File Reference	138
9.12.1	Detailed Description	139
9.13	object_vector.hh File Reference	139
9.13.1	Detailed Description	139
9.14	pointer_container.hh File Reference	139

9.14.1 Detailed Description	140
9.14.2 Macro Definition Documentation	140
9.14.2.1 JEOD_POINTER_CONTAINER	140
9.15 pointer_list.hh File Reference	140
9.15.1 Detailed Description	141
9.16 pointer_set.hh File Reference	141
9.16.1 Detailed Description	141
9.17 pointer_vector.hh File Reference	141
9.17.1 Detailed Description	142
9.18 primitive_container.hh File Reference	142
9.18.1 Detailed Description	142
9.18.2 Macro Definition Documentation	142
9.18.2.1 JEOD_PRIMITIVE_CONTAINER	143
9.19 primitive_list.hh File Reference	143
9.19.1 Detailed Description	143
9.20 primitive_serializer.cc File Reference	143
9.20.1 Detailed Description	144
9.21 primitive_serializer.hh File Reference	144
9.21.1 Detailed Description	144
9.22 primitive_set.hh File Reference	144
9.22.1 Detailed Description	145
9.23 primitive_vector.hh File Reference	145
9.23.1 Detailed Description	145
9.24 simple_checkpointable.hh File Reference	145
9.24.1 Detailed Description	146
Index	147

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Models	11
Utils	12
Container	13

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

jeod	Namespace jeod	25
----------------------	--------------------------	--------------------

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

ContainerType	
jeod::JeodContainer< ContainerType, ElemType >	44
jeod::JeodObjectContainer< ContainerType, ElemType >	63
jeod::JeodPrimitiveContainer< ContainerType, ElemType >	80
jeod::JeodContainer< ContainerType, ElemType *>	44
jeod::JeodPointerContainer< ContainerType, ElemType >	73
jeod::JeodCheckpointable	36
jeod::JeodContainer< ContainerType, ElemType >	44
jeod::SimpleCheckpointable	126
jeod::JeodContainer< ContainerType, ElemType *>	44
jeod::JeodObjectList< ElemType >	70
jeod::JeodObjectSet< ElemType >	71
jeod::JeodObjectVector< ElemType >	72
jeod::JeodPointerList< ElemType >	77
jeod::JeodPointerSet< ElemType >	78
jeod::JeodPointerVector< ElemType >	79
jeod::JeodPrimitiveList< ElemType >	84
jeod::JeodPrimitiveSerializerBase	90
jeod::JeodPrimitiveSerializer< Type >	85
jeod::JeodPrimitiveSerializer< ElemType >	85
jeod::JeodPrimitiveSet< ElemType >	94
jeod::JeodPrimitiveVector< ElemType >	95
jeod::JeodSTLContainer< ElemType, ContainerType >	108
jeod::JeodAssociativeContainer< ElemType, ContainerType >	27
jeod::JeodSequenceContainer< ElemType, ContainerType >	96
jeod::JeodSTLContainer< ElemType, std::list< ElemType > >	108
jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >	96
jeod::JeodList< ElemType >	54
jeod::JeodSTLContainer< ElemType, std::set< ElemType > >	108
jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >	27
jeod::JeodSet< ElemType >	104
jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >	108
jeod::JeodSequenceContainer< ElemType, std::vector< ElemType > >	96
jeod::JeodVector< ElemType >	121

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

jeod::JeodAssociativeContainer< ElemType, ContainerType >	27
This is the base class for the JEOD replacements of the STL associative containers	
jeod::JeodCheckpointable	36
A JeodCheckpointable is an object whose contents are opaque to Trick, and presumably other simulation engines, whose contents can nonetheless be checkpointed and restarted by using the methods defined herein	
jeod::JeodContainer< ContainerType, ElemType >	44
A JeodContainer is a JEOD STL sequence container replacement whose contents are checkpointable and restorable	
jeod::JeodList< ElemType >	54
The JEOD replacement for std::list	
jeod::JeodObjectContainer< ContainerType, ElemType >	63
A JeodObjectContainer is a JeodContainer that contains objects of type ElemType	
jeod::JeodObjectList< ElemType >	70
Defines a registry for defining a checkpointable list of objects	
jeod::JeodObjectSet< ElemType >	71
Defines a registry for defining a checkpointable set of objects	
jeod::JeodObjectVector< ElemType >	72
Defines a registry for defining a checkpointable vector of objects	
jeod::JeodPointerContainer< ContainerType, ElemType >	73
A JeodPointerContainer is a JeodContainer that contains pointers to objects of type ElemType	
jeod::JeodPointerList< ElemType >	77
Defines a registry for defining a checkpointable list of pointers	
jeod::JeodPointerSet< ElemType >	78
Defines a registry for defining a checkpointable set of pointers	
jeod::JeodPointerVector< ElemType >	79
Defines a registry for defining a checkpointable vector of pointers	
jeod::JeodPrimitiveContainer< ContainerType, ElemType >	80
A JeodPrimitiveContainer is a JeodContainer that contains primitive data of type ElemType	
jeod::JeodPrimitiveList< ElemType >	84
Defines a registry for defining a checkpointable list of primitives	
jeod::JeodPrimitiveSerializer< Type >	85
Serializer / deserializer for primitive data	
jeod::JeodPrimitiveSerializerBase	90
Base class for serializing / deserializing primitive data	

jeod::JeodPrimitiveSet< ElemType >	
Defines a registry for defining a checkpointable set of primitives	94
jeod::JeodPrimitiveVector< ElemType >	
Defines a registry for defining a checkpointable vector of primitives	95
jeod::JeodSequenceContainer< ElemType, ContainerType >	
This is the base class for the JEOD replacements of the STL sequence containers	96
jeod::JeodSet< ElemType >	
The JEOD replacement for std::set	104
jeod::JeodSTLContainer< ElemType, ContainerType >	
This is the base class for the JEOD replacements of the STL containers	108
jeod::JeodVector< ElemType >	
The JEOD replacement for std::vector	121
jeod::SimpleCheckpointable	
Simple checkpoint/restart interface by which an object can complete the restart process	126

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

checkpointable.hh	Define the class JeodCheckpointable, the base class for checkpointing and restoring data that are opaque to the simulation engine	131
container.hh	Define the class JeodContainer, which adds checkpointability to an STL sequence container replacement	131
jeod_associative_container.hh	Define checkpointable replacements for STL associative containers	132
jeod_container_compare.hh	Define comparison operators for JEOD STL container	132
jeod_list.hh	Define the class template JeodList	134
jeod_sequence_container.hh	Define checkpointable replacements for STL sequence containers	135
jeod_set.hh	Define the class template JeodSet	135
jeod_stl_container.hh	Define checkpointable replacements for STL containers	136
jeod_vector.hh	Define class template JeodVector	136
object_container.hh	Define class template JeodObjectContainer	137
object_list.hh	Define checkpointable replacements for STL sequence containers	138
object_set.hh	Define checkpointable replacements for STL associative containers	138
object_vector.hh	Define checkpointable replacements for STL sequence containers	139
pointer_container.hh	Define class template JeodPointerContainer	139
pointer_list.hh	Define checkpointable replacements for STL sequence containers	140
pointer_set.hh	Define checkpointable replacements for STL associative containers	141
pointer_vector.hh	Define checkpointable replacements for STL sequence containers	141

primitive_container.hh	
Define class template JeodPrimitiveContainer	142
primitive_list.hh	
Define checkpointable replacements for STL sequence containers	143
primitive_serializer.cc	
Define class JeodPrimitiveSerializerBase static methods	143
primitive_serializer.hh	
Define class template JeodPrimitiveSerializer	144
primitive_set.hh	
Define checkpointable replacements for STL associative containers	144
primitive_vector.hh	
Define checkpointable replacements for STL sequence containers	145
simple_checkpointable.hh	
Define the class SimpleCheckpointable	145

Chapter 6

Module Documentation

6.1 Models

Modules

- [Utils](#)

6.1.1 Detailed Description

6.2 Utils

Modules

- [Container](#)

6.2.1 Detailed Description

6.3 Container

Files

- file [checkpointable.hh](#)
Define the class JeodCheckpointable, the base class for checkpointing and restoring data that are opaque to the simulation engine.
- file [container.hh](#)
Define the class JeodContainer, which adds checkpointability to an STL sequence container replacement.
- file [jeod_associative_container.hh](#)
Define checkpointable replacements for STL associative containers.
- file [jeod_container_compare.hh](#)
Define comparison operators for JEOD STL container.
- file [jeod_list.hh](#)
Define the class template JeodList.
- file [jeod_sequence_container.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [jeod_set.hh](#)
Define the class template JeodSet.
- file [jeod_stl_container.hh](#)
Define checkpointable replacements for STL containers.
- file [jeod_vector.hh](#)
Define class template JeodVector.
- file [object_container.hh](#)
Define class template JeodObjectContainer.
- file [object_list.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [object_set.hh](#)
Define checkpointable replacements for STL associative containers.
- file [object_vector.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [pointer_container.hh](#)
Define class template JeodPointerContainer.
- file [pointer_list.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [pointer_set.hh](#)
Define checkpointable replacements for STL associative containers.
- file [pointer_vector.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [primitive_container.hh](#)
Define class template JeodPrimitiveContainer.
- file [primitive_list.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [primitive_serializer.hh](#)
Define class template JeodPrimitiveSerializer.
- file [primitive_set.hh](#)
Define checkpointable replacements for STL associative containers.
- file [primitive_vector.hh](#)
Define checkpointable replacements for STL sequence containers.
- file [simple_checkpointable.hh](#)
Define the class SimpleCheckpointable.
- file [primitive_serializer.cc](#)
Define class JeodPrimitiveSerializerBase static methods.

Namespaces

- [jeod](#)

Namespace jeod.

Macros

- `#define __USE_ISOC99`

Functions

- `template<typename ElemType , typename ContainerType >`
`bool operator< (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator< (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator< (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is greater than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than or equal to y.

- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is less than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than or equal to y.

6.3.1 Detailed Description

6.3.2 Macro Definition Documentation

6.3.2.1 __USE_ISOC99

```
#define __USE_ISOC99
```

Definition at line 23 of file primitive_serializer.cc.

6.3.3 Function Documentation

6.3.3.1 operator!=() [1/3]

```
template<typename ElemType , typename ContainerType >
bool operator!= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is not equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \neq y$

Definition at line 329 of file jeod_container_compare.hh.

6.3.3.2 operator"!="() [2/3]

```
template<typename ElemType , typename ContainerType >
bool operator!= (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is not equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \neq y$

Definition at line 341 of file jeod_container_compare.hh.

6.3.3.3 operator"!="() [3/3]

```
template<typename ElemType , typename ContainerType >
bool operator!= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is not equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \neq y$

Definition at line 353 of file jeod_container_compare.hh.

6.3.3.4 operator<>() [1/3]

```
template<typename ElemType , typename ContainerType >
bool operator< (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is less than y.

Parameters

x	Comparand
y	Comparand

Returns

$x < y$

Definition at line 181 of file jeod_container_compare.hh.

6.3.3.5 operator<>() [2/3]

```
template<typename ElemType , typename ContainerType >
bool operator< (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is less than y.

Parameters

x	Comparand
y	Comparand

Returns

$x < y$

Definition at line 193 of file jeod_container_compare.hh.

6.3.3.6 operator<() [3/3]

```
template<typename ElemType , typename ContainerType >
bool operator< (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is less than y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x < y$

Definition at line 205 of file jeod_container_compare.hh.

6.3.3.7 operator<=() [1/3]

```
template<typename ElemType , typename ContainerType >
bool operator<= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is less than or equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \leq y$

Definition at line 366 of file jeod_container_compare.hh.

6.3.3.8 operator<=() [2/3]

```
template<typename ElemType , typename ContainerType >
bool operator<= (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is less than or equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \leq y$

Definition at line 378 of file jeod_container_compare.hh.

6.3.3.9 operator<=() [3/3]

```
template<typename ElemType , typename ContainerType >
bool operator<= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is less than or equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \leq y$

Definition at line 390 of file jeod_container_compare.hh.

6.3.3.10 operator==([1/3]

```
template<typename ElemType , typename ContainerType >
bool operator==(
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

`x == y`

Definition at line 218 of file `jeod_container_compare.hh`.

6.3.3.11 operator==([2/3]

```
template<typename ElemType , typename ContainerType >
bool operator== (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is equal to y.

Parameters

<code>x</code>	Comparand
<code>y</code>	Comparand

Returns

`x == y`

Definition at line 230 of file `jeod_container_compare.hh`.

6.3.3.12 operator==([3/3]

```
template<typename ElemType , typename ContainerType >
bool operator== (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is equal to y.

Parameters

<code>x</code>	Comparand
<code>y</code>	Comparand

Returns

`x == y`

Definition at line 242 of file `jeod_container_compare.hh`.

6.3.3.13 `operator>()` [1/3]

```
template<typename ElemType , typename ContainerType >
bool operator> (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is greater than y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

`x > y`

Definition at line 255 of file jeod_container_compare.hh.

6.3.3.14 `operator>()` [2/3]

```
template<typename ElemType , typename ContainerType >
bool operator> (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is greater than y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

`x > y`

Definition at line 267 of file jeod_container_compare.hh.

6.3.3.15 `operator>()` [3/3]

```
template<typename ElemType , typename ContainerType >
bool operator> (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is greater than y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x > y$

Definition at line 279 of file jeod_container_compare.hh.

6.3.3.16 operator>=() [1/3]

```
template<typename ElemType , typename ContainerType >
bool operator>= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const ContainerType & y ) [inline]
```

Test if x is greater than or equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \geq y$

Definition at line 292 of file jeod_container_compare.hh.

6.3.3.17 operator>=() [2/3]

```
template<typename ElemType , typename ContainerType >
bool operator>= (
    const ContainerType & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is greater than or equal to y.

Parameters

<i>x</i>	Comparand
<i>y</i>	Comparand

Returns

$x \geq y$

Definition at line 304 of file jeod_container_compare.hh.

6.3.3.18 operator>=() [3/3]

```
template<typename ElemType , typename ContainerType >
bool operator>= (
    const jeod::JeodSTLContainer< ElemType, ContainerType > & x,
    const jeod::JeodSTLContainer< ElemType, ContainerType > & y ) [inline]
```

Test if x is greater than or equal to y.

Parameters

x	Comparand
y	Comparand

Returns

$x \geq y$

Definition at line 316 of file jeod_container_compare.hh.

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class [JeodAssociativeContainer](#)
This is the base class for the JEOD replacements of the STL associative containers.
- class [JeodCheckpointable](#)
A [JeodCheckpointable](#) is an object whose contents are opaque to Trick, and presumably other simulation engines, whose contents can nonetheless be checkpointed and restarted by using the methods defined herein.
- class [JeodContainer](#)
A [JeodContainer](#) is a JEOD STL sequence container replacement whose contents are checkpointable and restorable.
- class [JeodList](#)
The JEOD replacement for `std::list`.
- class [JeodObjectContainer](#)
A [JeodObjectContainer](#) is a [JeodContainer](#) that contains objects of type `ElemType`.
- class [JeodObjectList](#)
Defines a registry for defining a checkpointable list of objects.
- class [JeodObjectSet](#)
Defines a registry for defining a checkpointable set of objects.
- class [JeodObjectVector](#)
Defines a registry for defining a checkpointable vector of objects.
- class [JeodPointerContainer](#)
A [JeodPointerContainer](#) is a [JeodContainer](#) that contains pointers to objects of type `ElemType`.
- class [JeodPointerList](#)
Defines a registry for defining a checkpointable list of pointers.
- class [JeodPointerSet](#)
Defines a registry for defining a checkpointable set of pointers.
- class [JeodPointerVector](#)
Defines a registry for defining a checkpointable vector of pointers.
- class [JeodPrimitiveContainer](#)
A [JeodPrimitiveContainer](#) is a [JeodContainer](#) that contains primitive data of type `ElemType`.
- class [JeodPrimitiveList](#)

- Defines a registry for defining a checkpointable list of primitives.*
- class [JeodPrimitiveSerializer](#)
 - Serializer / deserializer for primitive data.*
- class [JeodPrimitiveSerializerBase](#)
 - Base class for serializing / deserializing primitive data.*
- class [JeodPrimitiveSet](#)
 - Defines a registry for defining a checkpointable set of primitives.*
- class [JeodPrimitiveVector](#)
 - Defines a registry for defining a checkpointable vector of primitives.*
- class [JeodSequenceContainer](#)
 - This is the base class for the JEOD replacements of the STL sequence containers.*
- class [JeodSet](#)
 - The JEOD replacement for `std::set`.*
- class [JeodSTLContainer](#)
 - This is the base class for the JEOD replacements of the STL containers.*
- class [JeodVector](#)
 - The JEOD replacement for `std::vector`.*
- class [SimpleCheckpointable](#)
 - The [SimpleCheckpointable](#) class provides a simple checkpoint/restart interface by which an object can complete the restart process.*

7.1.1 Detailed Description

Namespace `jeod`.

Chapter 8

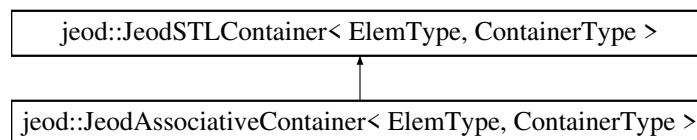
Data Structure Documentation

8.1 jeod::JeodAssociativeContainer< ElemType, ContainerType > Class Template Reference

This is the base class for the JEOD replacements of the STL associative containers.

```
#include <jeod_associative_container.hh>
```

Inheritance diagram for jeod::JeodAssociativeContainer< ElemType, ContainerType >:



Public Types

- using [this_container_type](#) = [JeodAssociativeContainer](#)< ElemType, ContainerType >
This type.
- using [base_container_type](#) = [JeodSTLContainer](#)< ElemType, ContainerType >
The [JeodSTLContainer](#).
- using [key_type](#) = typename ContainerType::key_type
Import the ContainerType::key_type.
- using [key_compare](#) = typename ContainerType::key_compare
Import the ContainerType::key_compare.
- using [value_compare](#) = typename ContainerType::value_compare
Import the ContainerType::value_compare.

Public Member Functions

- virtual `~JeodAssociativeContainer ()`=default
Destructor.
- `key_compare key_comp ()` const
Returns the key comparison object used to populate the contents.
- `value_compare value_comp ()` const
Returns the value comparison object used to populate the contents.
- `base_container_type::size_type count (const key_type &x)` const
Find the number of occurrences of the specified element.
- `base_container_type::iterator find (const key_type &x)`
Find the element specified by the given key.
- `base_container_type::const_iterator find (const key_type &x)` const
Find the element specified by the given key.
- `base_container_type::iterator lower_bound (const key_type &x)`
Find the start of a sequence specified by the given key.
- `base_container_type::const_iterator lower_bound (const key_type &x)` const
Find the start of a sequence specified by the given key.
- `base_container_type::iterator upper_bound (const key_type &x)`
Find the end of a sequence specified by the given key.
- `base_container_type::const_iterator upper_bound (const key_type &x)` const
Find the end of a sequence specified by the given key.
- `std::pair< typename base_container_type::iterator, typename base_container_type::iterator > equal_range (const key_type &x)`
Find the start and end of a sequence specified by the given key.
- `std::pair< typename base_container_type::const_iterator, typename base_container_type::const_iterator > equal_range (const key_type &x)` const
Find the start and end of a sequence specified by the given key.
- `template<class InputIterator >`
void `insert (InputIterator first, InputIterator last)`
Insert elements, initializing the inserted elements from the values pointed to by an iterator.
- `std::pair< typename base_container_type::iterator, bool > insert (const typename base_container_type::value_type &new_elem)`
Inserts the provided value into the associative list.
- void `erase (typename base_container_type::iterator position)`
Erase one item.
- void `erase (typename base_container_type::iterator first, typename base_container_type::iterator last)`
Erase a sequence of items.
- `base_container_type::size_type erase (const key_type &x)`
Erases the item(s) specified by supplied key from the contents.
- `iterator insert (iterator position, const value_type &new_elem)`
Insert a new element initialized with new_elem before the iterator position.

Protected Member Functions

- `JeodAssociativeContainer ()`=default
Default constructor.
- `JeodAssociativeContainer (const this_container_type &src)`
Copy constructor.
- `JeodAssociativeContainer (const ContainerType &src)`
Copy constructor from STL container.

Additional Inherited Members

8.1.1 Detailed Description

```
template<typename ElemType, typename ContainerType>
class jeod::JeodAssociativeContainer< ElemType, ContainerType >
```

This is the base class for the JEOD replacements of the STL associative containers.

The class derives from [JeodSTLContainer](#), the base class for the JEOD replacements of the STL containers.

A key goal of the JEOD STL associative container replacement effort is to provide checkpointable replacements that transparently provide the full functionality of the ISO/IEC 14882:2003 STL associative containers. This class begins that effort by defining types and member functions common to the STL set and map class templates. Non-common methods are the responsibility of derived class templates specialized to a specific container types.

Note

Exceptions to full functionality goal: The above goal is not and never will be fully achieved. Exceptions are:

- The full set of STL associative container constructors is not supplied.

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

8.1.2 Member Typedef Documentation

8.1.2.1 base_container_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodAssociativeContainer< ElemType, ContainerType >::base_container_type = JeodSTLContainer<Elem
Type, ContainerType>
```

The [JeodSTLContainer](#).

Definition at line 114 of file `jeod_associative_container.hh`.

8.1.2.2 key_compare

```
template<typename ElemType, typename ContainerType>
using jeod::JeodAssociativeContainer< ElemType, ContainerType >::key_compare = typename ContainerType::key_compare
```

Import the `ContainerType::key_compare`.

Definition at line 124 of file `jeod_associative_container.hh`.

8.1.2.3 key_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodAssociativeContainer< ElemType, ContainerType >::key_type = typename ContainerType::key_type
```

Import the ContainerType::key_type.

Definition at line 119 of file jeod_associative_container.hh.

8.1.2.4 this_container_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodAssociativeContainer< ElemType, ContainerType >::this_container_type = JeodAssociativeContainer< ElemType, ContainerType>
```

This type.

Definition at line 109 of file jeod_associative_container.hh.

8.1.2.5 value_compare

```
template<typename ElemType, typename ContainerType>
using jeod::JeodAssociativeContainer< ElemType, ContainerType >::value_compare = typename ContainerType::value_compare
```

Import the ContainerType::value_compare.

Definition at line 129 of file jeod_associative_container.hh.

8.1.3 Constructor & Destructor Documentation

8.1.3.1 ~JeodAssociativeContainer()

```
template<typename ElemType, typename ContainerType>
virtual jeod::JeodAssociativeContainer< ElemType, ContainerType >::~~JeodAssociativeContainer
( ) [virtual], [default]
```

Destructor.

8.1.3.2 JeodAssociativeContainer() [1/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::JeodAssociativeContainer ( ) [protected],
[default]
```

Default constructor.

Note: Making this protected precludes someone from declaring an object to be of type JEODSTLContainer. Access is via some other class that inherits from this class.

8.1.3.3 JeodAssociativeContainer() [2/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::JeodAssociativeContainer (
    const this_container_type & src ) [inline], [protected]
```

Copy constructor.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 318 of file jeod_associative_container.hh.

8.1.3.4 JeodAssociativeContainer() [3/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::JeodAssociativeContainer (
    const ContainerType & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 327 of file jeod_associative_container.hh.

8.1.4 Member Function Documentation

8.1.4.1 count()

```
template<typename ElemType, typename ContainerType>
base_container_type::size_type jeod::JeodAssociativeContainer< ElemType, ContainerType >↔
::count (
    const key_type & x ) const [inline]
```

Find the number of occurrences of the specified element.

Definition at line 175 of file jeod_associative_container.hh.

8.1.4.2 equal_range() [1/2]

```
template<typename ElemType, typename ContainerType>
std::pair<typename base_container_type::iterator, typename base_container_type::iterator>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::equal_range (
    const key_type & x ) [inline]
```

Find the start and end of a sequence specified by the given key.

Definition at line 231 of file jeod_associative_container.hh.

8.1.4.3 equal_range() [2/2]

```
template<typename ElemType, typename ContainerType>
std::pair<typename base_container_type::const_iterator, typename base_container_type::const_iterator>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::equal_range (
    const key_type & x ) const [inline]
```

Find the start and end of a sequence specified by the given key.

Definition at line 240 of file jeod_associative_container.hh.

8.1.4.4 erase() [1/3]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodAssociativeContainer< ElemType, ContainerType >::erase (
    typename base_container_type::iterator position ) [inline]
```

Erase one item.

Parameters

<i>position</i>	Position to be erased
-----------------	-----------------------

Definition at line 277 of file jeod_associative_container.hh.

8.1.4.5 erase() [2/3]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodAssociativeContainer< ElemType, ContainerType >::erase (
    typename base_container_type::iterator first,
    typename base_container_type::iterator last ) [inline]
```

Erase a sequence of items.

Parameters

<i>first</i>	First element to be erased
<i>last</i>	One past last element to be erased

Definition at line 287 of file jeod_associative_container.hh.

8.1.4.6 erase() [3/3]

```
template<typename ElemType, typename ContainerType>
base_container_type::size_type jeod::JeodAssociativeContainer< ElemType, ContainerType >↵
::erase (
    const key_type & x ) [inline]
```

Erases the item(s) specified by supplied key from the contents.

Parameters

<i>x</i>	Key of item(s) to be erased
----------	-----------------------------

Definition at line 296 of file jeod_associative_container.hh.

8.1.4.7 find() [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::iterator jeod::JeodAssociativeContainer< ElemType, ContainerType >::find
(
    const key_type & x ) [inline]
```

Find the element specified by the given key.

Definition at line 183 of file jeod_associative_container.hh.

8.1.4.8 find() [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::const_iterator jeod::JeodAssociativeContainer< ElemType, ContainerType
>::find (
    const key_type & x ) const    [inline]
```

Find the element specified by the given key.

Definition at line 191 of file jeod_associative_container.hh.

8.1.4.9 insert() [1/3]

```
template<typename ElemType, typename ContainerType>
iterator jeod::JeodSTLContainer< ElemType, ContainerType >::insert    [inline]
```

Insert a new element initialized with *new_elem* before the iterator *position*.

Parameters

<i>position</i>	Insertion position
<i>new_elem</i>	Element value to be inserted

Returns

Iterator that points to the newly-inserted element

Definition at line 340 of file jeod_stl_container.hh.

8.1.4.10 insert() [2/3]

```
template<typename ElemType, typename ContainerType>
template<class InputIterator >
void jeod::JeodAssociativeContainer< ElemType, ContainerType >::insert (
    InputIterator first,
    InputIterator last )    [inline]
```

Insert elements, initializing the inserted elements from the values pointed to by an iterator.

Parameters

<i>first</i>	Input iterator
<i>last</i>	Input iterator

Definition at line 258 of file jeod_associative_container.hh.

8.1.4.11 insert() [3/3]

```
template<typename ElemType, typename ContainerType>
std::pair<typename base_container_type::iterator, bool> jeod::JeodAssociativeContainer< ElemType, ContainerType >::insert (
    const typename base_container_type::value_type & new_elem ) [inline]
```

Inserts the provided value into the associative list.

Parameters

<i>new_elem</i>	Element value to be inserted
-----------------	------------------------------

Definition at line 267 of file jeod_associative_container.hh.

8.1.4.12 key_comp()

```
template<typename ElemType, typename ContainerType>
key_compare jeod::JeodAssociativeContainer< ElemType, ContainerType >::key_comp ( ) const
[inline]
```

Returns the key comparison object used to populate the contents.

Definition at line 151 of file jeod_associative_container.hh.

8.1.4.13 lower_bound() [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::iterator jeod::JeodAssociativeContainer< ElemType, ContainerType >::lower_bound (
    const key_type & x ) [inline]
```

Find the start of a sequence specified by the given key.

Definition at line 199 of file jeod_associative_container.hh.

8.1.4.14 lower_bound() [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::const_iterator jeod::JeodAssociativeContainer< ElemType, ContainerType >::lower_bound (
    const key_type & x ) const [inline]
```

Find the start of a sequence specified by the given key.

Definition at line 207 of file jeod_associative_container.hh.

8.1.4.15 `upper_bound()` [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::iterator jeod::JeodAssociativeContainer< ElemType, ContainerType >↔
::upper_bound (
    const key_type & x ) [inline]
```

Find the end of a sequence specified by the given key.

Definition at line 215 of file `jeod_associative_container.hh`.

8.1.4.16 `upper_bound()` [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::const_iterator jeod::JeodAssociativeContainer< ElemType, ContainerType
>::upper_bound (
    const key_type & x ) const [inline]
```

Find the end of a sequence specified by the given key.

Definition at line 223 of file `jeod_associative_container.hh`.

8.1.4.17 `value_comp()`

```
template<typename ElemType, typename ContainerType>
value_compare jeod::JeodAssociativeContainer< ElemType, ContainerType >::value_comp ( ) const
[inline]
```

Returns the value comparison object used to populate the contents.

Definition at line 159 of file `jeod_associative_container.hh`.

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

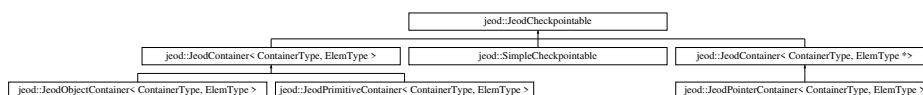
- [jeod_associative_container.hh](#)

8.2 `jeod::JeodCheckpointable` Class Reference

A `JeodCheckpointable` is an object whose contents are opaque to Trick, and presumably other simulation engines, whose contents can nonetheless be checkpointed and restarted by using the methods defined herein.

```
#include <checkpointable.hh>
```

Inheritance diagram for `jeod::JeodCheckpointable`:



Public Member Functions

- [JeodCheckpointable](#) ()=default
- virtual [~JeodCheckpointable](#) ()=default
- [JeodCheckpointable](#) (const [JeodCheckpointable](#) &)=delete
- [JeodCheckpointable](#) & operator= (const [JeodCheckpointable](#) &)=delete
- virtual void [pre_checkpoint](#) ()

In general, perform object-specific operations that need to be performed in anticipation of a checkpoint, typically allocating and populating memory.
- virtual void [post_checkpoint](#) ()

In general, perform object-specific operations that need to be performed after checkpoint completion, typically freeing memory used for checkpointing.
- virtual void [pre_restart](#) ()

In general, perform object-specific operations that need to be performed in anticipation of a restart, typically releasing resources.
- virtual void [post_restart](#) ()

In general, perform object-specific operations that need to be performed after restart completion.
- virtual void [initialize_checkpointable](#) (const void *container, const std::type_info &container_type, const std::string &elem_name)

In general, perform initialization actions such as obtaining requisite type information, registering Checkpointable objects contained within the object, etc.
- virtual void [undo_initialize_checkpointable](#) (const void *container, const std::type_info &container_type, const std::string &elem_name)

In general, undo external actions performed by initialize_checkpointable.
- virtual const std::string [get_init_value](#) ()

In general, return the value of the initialization action.
- virtual const std::string [get_final_name](#) ()

In general, return the name of the finalization action.
- virtual const std::string [get_final_value](#) ()

In general, return the value of the finalization action.
- virtual void [start_checkpoint](#) ()=0

Prepare to checkpoint the object in question.
- virtual void [advance_checkpoint](#) ()=0

Advance to the next item to be checkpointed.
- virtual bool [is_checkpoint_finished](#) ()=0

Return true if all contents have been checkpointed, false otherwise.
- virtual const std::string [get_init_name](#) ()=0

Return the name of the action, if any, that will be performed prior to performing the individual actions.
- virtual const std::string [get_item_name](#) ()=0

Return the name of the action that will restore the value at the current checkpoint position.
- virtual const std::string [get_item_value](#) ()=0

Return the value of the item to be written to the checkpoint file.
- virtual int [perform_restore_action](#) (const std::string &action_name, const std::string &action_value)=0

Perform a checkpoint-restart action that will, in part, restore the object to its state at the time of the checkpoint.

Friends

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

8.2.1 Detailed Description

A [JeodCheckpointable](#) is an object whose contents are opaque to Trick, and presumably other simulation engines, whose contents can nonetheless be checkpointed and restarted by using the methods defined herein.

Definition at line 78 of file checkpointable.hh.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 JeodCheckpointable() [1/2]

```
jeod::JeodCheckpointable::JeodCheckpointable ( ) [default]
```

8.2.2.2 ~JeodCheckpointable()

```
virtual jeod::JeodCheckpointable::~~JeodCheckpointable ( ) [virtual], [default]
```

8.2.2.3 JeodCheckpointable() [2/2]

```
jeod::JeodCheckpointable::JeodCheckpointable (
    const JeodCheckpointable & ) [delete]
```

8.2.3 Member Function Documentation

8.2.3.1 advance_checkpoint()

```
virtual void jeod::JeodCheckpointable::advance_checkpoint ( ) [pure virtual]
```

Advance to the next item to be checkpointed.

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), [jeod::JeodObjectContainer< ContainerType, ElemType >](#), and [jeod::SimpleCheckpointable](#).

8.2.3.2 get_final_name()

```
const std::string jeod::JeodCheckpointable::get_final_name ( ) [inline], [virtual]
```

In general, return the name of the finalization action.

The returned value is written to the checkpoint file as the name of the final action, but only if this name is not empty.

The default implementation is the empty string.

Reimplemented in [jeod::JeodContainer< ContainerType, ElemType >](#), and [jeod::JeodContainer< ContainerType, ElemType *>](#).

Definition at line 190 of file checkpointable.hh.

8.2.3.3 get_final_value()

```
const std::string jeod::JeodCheckpointable::get_final_value ( ) [inline], [virtual]
```

In general, return the value of the finalization action.

The returned value is written to the checkpoint file as the argument of the final action, but only if the finalization action is not empty.

The default implementation is the empty string.

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 202 of file checkpointable.hh.

8.2.3.4 get_init_name()

```
virtual const std::string jeod::JeodCheckpointable::get_init_name ( ) [pure virtual]
```

Return the name of the action, if any, that will be performed prior to performing the individual actions.

Note: The init name must be alphanumeric or empty.

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), and [jeod::SimpleCheckpointable](#).

8.2.3.5 get_init_value()

```
const std::string jeod::JeodCheckpointable::get_init_value ( ) [inline], [virtual]
```

In general, return the value of the initialization action.

The returned value is written to the checkpoint file as the argument of the init action, but only if the initialization action is not empty.

The default implementation is the empty string.

Definition at line 178 of file checkpointable.hh.

8.2.3.6 `get_item_name()`

```
virtual const std::string jeod::JeodCheckpointable::get_item_name ( ) [pure virtual]
```

Return the name of the action that will restore the value at the current checkpoint position.

This action name and the corresponding value will be written to the checkpoint file in the form "owner.action(value);".

Note: The item name must be alphanumeric.

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), and [jeod::SimpleCheckpointable](#).

8.2.3.7 `get_item_value()`

```
virtual const std::string jeod::JeodCheckpointable::get_item_value ( ) [pure virtual]
```

Return the value of the item to be written to the checkpoint file.

Translation of the true value to a string is up to the implementation. The string value must be something that the `restore_perform_action` method can translate back to the true value and should also be human-readable; people as well as the Memory Manager read checkpoint files.

Implemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#), [jeod::JeodPointerContainer< ContainerType, ElemType >](#), [jeod::JeodPrimitiveContainer< ContainerType, ElemType >](#), and [jeod::SimpleCheckpointable](#).

8.2.3.8 `initialize_checkpointable()`

```
void jeod::JeodCheckpointable::initialize_checkpointable (
    const void * container,
    const std::type_info & container_type,
    const std::string & elem_name ) [inline], [virtual]
```

In general, perform initialization actions such as obtaining requisite type information, registering Checkpointable objects contained within the object, etc.

The default implementation is to do nothing.

Parameters

<i>container</i>	The object that contains this object.
<i>container_type</i>	The type of the containing object.
<i>elem_name</i>	The name of the this object in the containing object.

Reimplemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), and [jeod::JeodPointerContainer< ContainerType, ElemType >](#).

Definition at line 265 of file `checkpointable.hh`.

8.2.3.9 is_checkpoint_finished()

```
virtual bool jeod::JeodCheckpointable::is_checkpoint_finished ( ) [pure virtual]
```

Return true if all contents have been checkpointed, false otherwise.

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), and [jeod::SimpleCheckpointable](#).

8.2.3.10 operator=()

```
JeodCheckpointable& jeod::JeodCheckpointable::operator= (
    const JeodCheckpointable & ) [delete]
```

8.2.3.11 perform_restore_action()

```
virtual int jeod::JeodCheckpointable::perform_restore_action (
    const std::string & action_name,
    const std::string & action_value ) [pure virtual]
```

Perform a checkpoint-restart action that will, in part, restore the object to its state at the time of the checkpoint.

The method is called for each entry in the checkpoint file that pertains to this object.

Parameters

<i>action_name</i>	The name of the action.
<i>action_value</i>	The value of the action.

Returns

Success (zero) / failure (non-zero).

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), and [jeod::SimpleCheckpointable](#).

8.2.3.12 post_checkpoint()

```
void jeod::JeodCheckpointable::post_checkpoint ( ) [inline], [virtual]
```

In general, perform object-specific operations that need to be performed after checkpoint completion, typically freeing memory used for checkpointing.

The simulation engine calls this method after checkpoint-proper completion.

The default implementation is to do nothing.

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 226 of file checkpointable.hh.

8.2.3.13 post_restart()

```
void jeod::JeodCheckpointable::post_restart ( ) [inline], [virtual]
```

In general, perform object-specific operations that need to be performed after restart completion.

The default implementation is to do nothing.

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 249 of file checkpointable.hh.

8.2.3.14 pre_checkpoint()

```
void jeod::JeodCheckpointable::pre_checkpoint ( ) [inline], [virtual]
```

In general, perform object-specific operations that need to be performed in anticipation of a checkpoint, typically allocating and populating memory.

The simulation engine calls this method prior to checkpointing allocations.

The default implementation is to do nothing.

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 214 of file checkpointable.hh.

8.2.3.15 pre_restart()

```
void jeod::JeodCheckpointable::pre_restart ( ) [inline], [virtual]
```

In general, perform object-specific operations that need to be performed in anticipation of a restart, typically releasing resources.

The simulation engine calls this method prior to restoring allocated data.

The default implementation is to do nothing.

Definition at line 238 of file checkpointable.hh.

8.2.3.16 start_checkpoint()

```
virtual void jeod::JeodCheckpointable::start_checkpoint ( ) [pure virtual]
```

Prepare to checkpoint the object in question.

Implemented in [jeod::JeodContainer< ContainerType, ElemType >](#), [jeod::JeodContainer< ContainerType, ElemType *>](#), [jeod::JeodObjectContainer< ContainerType, ElemType >](#), and [jeod::SimpleCheckpointable](#).

8.2.3.17 undo_initialize_checkpointable()

```
void jeod::JeodCheckpointable::undo_initialize_checkpointable (
    const void * container,
    const std::type_info & container_type,
    const std::string & elem_name ) [inline], [virtual]
```

In general, undo external actions performed by `initialize_checkpointable`.

The default implementation is to do nothing.

Parameters

<i>container</i>	The object that contains this object.
<i>container_type</i>	The type of the containing object.
<i>elem_name</i>	The name of the this object in the containing object.

Definition at line 281 of file `checkpointable.hh`.

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

```
void init_attrjeod__JeodCheckpointable ( ) [friend]
```

8.2.4.2 InputProcessor

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

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

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

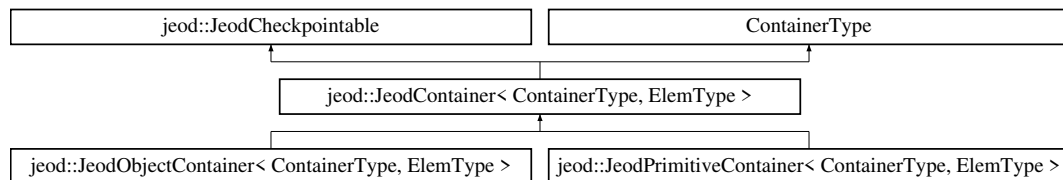
- [checkpointable.hh](#)

8.3 jeod::JeodContainer< ContainerType, ElemType > Class Template Reference

A [JeodContainer](#) is a JEOD STL sequence container replacement whose contents are checkpointable and restorable.

```
#include <container.hh>
```

Inheritance diagram for `jeod::JeodContainer< ContainerType, ElemType >`:



Public Types

- using `this_container_type = JeodContainer< ContainerType, ElemType >`
This particular [JeodContainer](#) type.
- using `stl_container_type = typename ContainerType::stl_container_type`
Import the ContainerType's container type.

Public Member Functions

- [JeodContainer](#) ()
Default constructor.
- [JeodContainer](#) (const [this_container_type](#) &source)
Copy constructor.
- [JeodContainer](#) (const [stl_container_type](#) &source)
Copy constructor.
- [JeodContainer](#) & operator= (const [this_container_type](#) &source)
Assignment operator.
- [JeodContainer](#) & operator= (const [stl_container_type](#) &source)
Assignment operator.
- [~JeodContainer](#) () override=default
Destructor.
- void [swap_contents](#) ([this_container_type](#) &other)
Swap STL sequence container contents – but not the stuff related to checkpoint or restart.
- void [swap_contents](#) ([stl_container_type](#) &other)
Swap STL sequence container contents – but not the stuff related to checkpoint or restart.
- virtual void [perform_insert_action](#) (const std::string &value)=0
Push a value onto the end of the contents.
- virtual void [perform_cleanup_action](#) (const std::string &value)
Cleanup detritus created during the restore process.
- void [initialize_checkpointable](#) (const void *container, const std::type_info &container_type, const std::string &elem_name) override
Initialize a checkpointable object, called by the checkpoint manager.
- void [start_checkpoint](#) () override
Prepare to checkpoint the object.

- void [advance_checkpoint](#) () override
Advance to the next item to be checkpointed.
- bool [is_checkpoint_finished](#) () override
Indicate whether the checkpoint dump of this object is finished.
- const std::string [get_init_name](#) () override
Names the action to be performed prior to performing any of the restore actions.
- const std::string [get_item_name](#) () override
Return the name of the action to be printed along with the current value.
- const std::string [get_final_name](#) () override
Names the action to be performed after to performing any of the restore actions.
- int [perform_restore_action](#) (const std::string &action_name, const std::string &action_value) override
Perform a checkpoint-restart action that will, in part, restore the object to its state at the time of the checkpoint.

Protected Attributes

- ContainerType::iterator [checkpoint_iter](#)
Iterator for walking through the container during checkpoint.
- const JeodMemoryTypeDescriptor * [elem_type_descriptor](#) {}
Memory model descriptor of the type of data stored in the container.

Friends

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

8.3.1 Detailed Description

```
template<typename ContainerType, typename ElemType>
class jeod::JeodContainer< ContainerType, ElemType >
```

A [JeodContainer](#) is a JEOD STL sequence container replacement whose contents are checkpointable and restorable.

Definition at line 82 of file container.hh.

8.3.2 Member Typedef Documentation

8.3.2.1 stl_container_type

```
template<typename ContainerType, typename ElemType>
using jeod::JeodContainer< ContainerType, ElemType >::stl\_container\_type = typename ContainerType::stl_container_type
```

Import the ContainerType's container type.

Definition at line 96 of file container.hh.

8.3.2.2 `this_container_type`

```
template<typename ContainerType, typename ElemType>
using jeod::JeodContainer< ContainerType, ElemType >::this_container_type = JeodContainer<ContainerType, ElemType>
```

This particular `JeodContainer` type.

Definition at line 91 of file container.hh.

8.3.3 Constructor & Destructor Documentation

8.3.3.1 `JeodContainer()` [1/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodContainer< ContainerType, ElemType >::JeodContainer ( ) [inline]
```

Default constructor.

Definition at line 103 of file container.hh.

8.3.3.2 `JeodContainer()` [2/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodContainer< ContainerType, ElemType >::JeodContainer (
    const this_container_type & source ) [inline]
```

Copy constructor.

Note

This copies the source's `ContainerType` contents only. The `Checkpointable` contents and the added checkpoint members are not copied.

Parameters

<code>source</code>	Container to be copied.
---------------------	-------------------------

Definition at line 119 of file container.hh.

8.3.3.3 JeodContainer() [3/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodContainer< ContainerType, ElemType >::JeodContainer (
    const stl_container_type & source ) [inline]
```

Copy constructor.

Note

This copies the source's ContainerType contents only. The Checkpointable contents and the added checkpoint members are not copied.

Parameters

<i>source</i>	Container to be copied.
---------------	-------------------------

Definition at line 135 of file container.hh.

8.3.3.4 ~JeodContainer()

```
template<typename ContainerType, typename ElemType>
jeod::JeodContainer< ContainerType, ElemType >::~~JeodContainer ( ) [override], [default]
```

Destructor.

8.3.4 Member Function Documentation**8.3.4.1 advance_checkpoint()**

```
template<typename ContainerType, typename ElemType>
void jeod::JeodContainer< ContainerType, ElemType >::advance_checkpoint ( ) [inline], [override],
[virtual]
```

Advance to the next item to be checkpointed.

In the case of a [JeodContainer](#), this method simply advances the checkpoint iterator to point to the next item in the contents.

Implements [jeod::JeodCheckpointable](#).

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 251 of file container.hh.

8.3.4.2 `get_final_name()`

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodContainer< ContainerType, ElemType >::get_final_name ( ) [inline],
[override], [virtual]
```

Names the action to be performed after to performing any of the restore actions.

In the case of a [JeodContainer](#), the init name is always "cleanup".

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

Definition at line 293 of file container.hh.

8.3.4.3 `get_init_name()`

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodContainer< ContainerType, ElemType >::get_init_name ( ) [inline],
[override], [virtual]
```

Names the action to be performed prior to performing any of the restore actions.

In the case of a [JeodContainer](#), the init name is always "clear".

Implements [jeod::JeodCheckpointable](#).

Definition at line 272 of file container.hh.

8.3.4.4 `get_item_name()`

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodContainer< ContainerType, ElemType >::get_item_name ( ) [inline],
[override], [virtual]
```

Return the name of the action to be printed along with the current value.

In the case of a [JeodContainer](#), the action name is always "insert".

Implements [jeod::JeodCheckpointable](#).

Definition at line 282 of file container.hh.

8.3.4.5 initialize_checkpointable()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodContainer< ContainerType, ElemType >::initialize_checkpointable (
    const void * container,
    const std::type_info & container_type,
    const std::string & elem_name ) [inline], [override], [virtual]
```

Initialize a checkpointable object, called by the checkpoint manager.

In the case of a [JeodContainer](#), this method gets the descriptor for the type of data stored in the container.

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

Definition at line 224 of file container.hh.

Referenced by `jeod::JeodPointerContainer< ContainerType, ElemType >::initialize_checkpointable()`.

8.3.4.6 is_checkpoint_finished()

```
template<typename ContainerType, typename ElemType>
bool jeod::JeodContainer< ContainerType, ElemType >::is_checkpoint_finished ( ) [inline],
[override], [virtual]
```

Indicate whether the checkpoint dump of this object is finished.

In the case of a [JeodContainer](#), the dump is finished when the internal checkpoint iterator points beyond the last item in the contents.

Implements [jeod::JeodCheckpointable](#).

Definition at line 261 of file container.hh.

8.3.4.7 operator=() [1/2]

```
template<typename ContainerType, typename ElemType>
JeodContainer& jeod::JeodContainer< ContainerType, ElemType >::operator= (
    const this_container_type & source ) [inline]
```

Assignment operator.

Note

This copies the source's ContainerType contents only. The Checkpointable contents and the added checkpoint members are not copied.

Parameters

<i>source</i>	Container to be copied.
---------------	-------------------------

Definition at line 151 of file container.hh.

Referenced by `jeod::JeodPrimitiveContainer< ContainerType, ElemType >::operator=()`, and `jeod::JeodPointer< ContainerType, ElemType >::operator=()`.

8.3.4.8 operator=() [2/2]

```
template<typename ContainerType, typename ElemType>
JeodContainer& jeod::JeodContainer< ContainerType, ElemType >::operator= (
    const stl_container_type & source ) [inline]
```

Assignment operator.

Note

This copies the source's ContainerType contents only. The Checkpointable contents and the added checkpoint members are not copied.

Parameters

<i>source</i>	Container to be copied.
---------------	-------------------------

Definition at line 168 of file container.hh.

8.3.4.9 perform_cleanup_action()

```
template<typename ContainerType, typename ElemType>
virtual void jeod::JeodContainer< ContainerType, ElemType >::perform_cleanup_action (
    const std::string & value ) [inline], [virtual]
```

Cleanup detritus created during the restore process.

The default action is to do nothing.

Parameters

<i>value</i>	String name of cleanup target. This member should be protected or (even better) private. It is marked as public to avoid problems with Trick and SWIG.
--------------	--

Reimplemented in `jeod::JeodObjectContainer< ContainerType, ElemType >`.

Definition at line 216 of file container.hh.

8.3.4.10 perform_insert_action()

```
template<typename ContainerType, typename ElemType>
virtual void jeod::JeodContainer< ContainerType, ElemType >::perform_insert_action (
    const std::string & value ) [pure virtual]
```

Push a value onto the end of the contents.

This method is pure virtual because the value provided to the method is a string. Translating the input string to the appropriate element type is the responsibility of template instantiations.

Parameters

<i>value</i>	Value, in string form, to be added to the contents.
--------------	---

Note

This member should be protected or (even better) private. It is marked as public to avoid problems with Trick and SWIG.

Implemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#), [jeod::JeodPointerContainer< ContainerType, ElemType >](#) and [jeod::JeodPrimitiveContainer< ContainerType, ElemType >](#).

8.3.4.11 perform_restore_action()

```
template<typename ContainerType, typename ElemType>
int jeod::JeodContainer< ContainerType, ElemType >::perform_restore_action (
    const std::string & action_name,
    const std::string & action_value ) [inline], [override], [virtual]
```

Perform a checkpoint-restart action that will, in part, restore the object to its state at the time of the checkpoint.

In the case of a [JeodContainer](#), the actions are "clear", "insert", and "cleanup". The checkpoint writer automatically creates an initial "clear" entry as the first entry in the checkpoint file for a [JeodCheckpointable](#) object and a "cleanup" entry as the final entry. An "insert" entry is created for each element in the container's contents.

Implements [jeod::JeodCheckpointable](#).

Definition at line 308 of file container.hh.

8.3.4.12 start_checkpoint()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodContainer< ContainerType, ElemType >::start_checkpoint ( ) [inline], [override],
[virtual]
```

Prepare to checkpoint the object.

In the case of a [JeodContainer](#), this method initializes a checkpoint iterator, data member `checkpoint_iter`, to the start of the contents.

Implements [jeod::JeodCheckpointable](#).

Reimplemented in [jeod::JeodObjectContainer< ContainerType, ElemType >](#).

Definition at line 240 of file `container.hh`.

8.3.4.13 swap_contents() [1/2]

```
template<typename ContainerType, typename ElemType>
void jeod::JeodContainer< ContainerType, ElemType >::swap_contents (
    this_container_type & other ) [inline]
```

Swap STL sequence container contents – but not the stuff related to checkpoint or restart.

Definition at line 183 of file `container.hh`.

8.3.4.14 swap_contents() [2/2]

```
template<typename ContainerType, typename ElemType>
void jeod::JeodContainer< ContainerType, ElemType >::swap_contents (
    stl_container_type & other ) [inline]
```

Swap STL sequence container contents – but not the stuff related to checkpoint or restart.

Definition at line 192 of file `container.hh`.

8.3.5 Friends And Related Function Documentation**8.3.5.1 init_attrjeod__JeodContainer**

```
template<typename ContainerType, typename ElemType>
void init_attrjeod__JeodContainer ( ) [friend]
```

8.3.5.2 InputProcessor

```
template<typename ContainerType, typename ElemType>
friend class InputProcessor [friend]
```

Definition at line 85 of file container.hh.

8.3.6 Field Documentation

8.3.6.1 checkpoint_iter

```
template<typename ContainerType, typename ElemType>
ContainerType::iterator jeod::JeodContainer< ContainerType, ElemType >::checkpoint_iter [protected]
```

Iterator for walking through the container during checkpoint.

trick_io(**)

Definition at line 346 of file container.hh.

Referenced by jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value().

8.3.6.2 elem_type_descriptor

```
template<typename ContainerType, typename ElemType>
const JeodMemoryTypeDescriptor* jeod::JeodContainer< ContainerType, ElemType >::elem_type_↵
descriptor {} [protected]
```

Memory model descriptor of the type of data stored in the container.

trick_io(**)

Definition at line 351 of file container.hh.

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

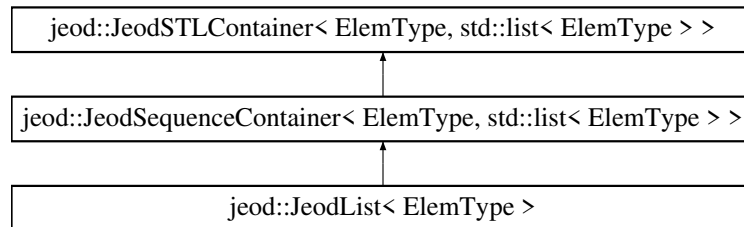
- [container.hh](#)

8.4 jeod::JeodList< ElemType > Class Template Reference

The JEOD replacement for `std::list`.

```
#include <jeod_list.hh>
```

Inheritance diagram for `jeod::JeodList< ElemType >`:



Public Types

- using `this_container_type` = `JeodList< ElemType >`
This particular `JeodList` type.
- using `jeod_sequence_container_type` = `JeodSequenceContainer< ElemType, std::list< ElemType > >`
The `JeodSequenceContainer` type.
- using `jeod_stl_container_type` = `JeodSTLContainer< ElemType, std::list< ElemType > >`
The `JeodSTLContainer` type.
- using `stl_container_type` = `std::list< ElemType >`
The `std::list` itself.

Public Member Functions

- virtual `~JeodList()`=default
Destructor.
- `JeodList & operator=` (const `this_container_type` &src)
Copy contents from the given source.
- `JeodList & operator=` (const `stl_container_type` &src)
Copy contents from the given source.
- void `merge` (`stl_container_type` &other)
Merge the contents of some other list into this list, emptying the other list.
- template<typename Compare >
void `merge` (`stl_container_type` &other, Compare comp)
Merge the contents of some other list into this list using the provided comparator to guide the merge.
- void `push_front` (const `ElemType` &elem)
Add an element to the head of the list.
- void `pop_front` ()
Deletes the element at the head of the list.
- void `remove` (const `ElemType` &value)
Remove elements from the list that are equal to the provided value.
- template<typename Predicate >
void `remove_if` (Predicate pred)
Remove elements from the list that pass the provided test.
- void `reverse` ()

Reverse the list.

- void `splice` (typename `jeod_stl_container_type::iterator` position, `stl_container_type` &other)

Inserts the contents of other before position, emptying other.

- void `splice` (typename `jeod_stl_container_type::iterator` position, `stl_container_type` &other, typename `jeod_stl_container_type::iterator` other_pos)

Inserts the element other_pos of other before position, deleting that element from other.

- void `splice` (typename `jeod_stl_container_type::iterator` position, `stl_container_type` &other, typename `jeod_stl_container_type::iterator` first, typename `jeod_stl_container_type::iterator` last)

Inserts elements in other from first up to but not including last before position, deleting those element from other.

- void `sort` ()

Sort using the default comparison operator.

- template<typename Compare >
void `sort` (Compare comp)

Sort using the provided comparator.

- void `unique` ()

Remove duplicates using the default equality operator.

- template<typename BinaryPredicate >
void `unique` (BinaryPredicate comp)

Remove duplicates using the provided comparator.

Protected Member Functions

- `JeodList` ()=default

Default constructor.

- `JeodList` (const `this_container_type` &src)

Copy constructor.

- `JeodList` (const `stl_container_type` &src)

Copy constructor from STL container.

Additional Inherited Members

8.4.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodList< ElemType >
```

The JEOD replacement for std::list.

Definition at line 91 of file jeod_list.hh.

8.4.2 Member Typedef Documentation

8.4.2.1 jeod_sequence_container_type

```
template<typename ElemType >
using jeod::JeodList< ElemType >::jeod_sequence_container_type = JeodSequenceContainer<ElemType, std::list<ElemType> >
```

The [JeodSequenceContainer](#) type.

Definition at line 104 of file jeod_list.hh.

8.4.2.2 jeod_stl_container_type

```
template<typename ElemType >
using jeod::JeodList< ElemType >::jeod_stl_container_type = JeodSTLContainer<ElemType, std::list<ElemType> >
```

The [JeodSTLContainer](#) type.

Definition at line 109 of file jeod_list.hh.

8.4.2.3 stl_container_type

```
template<typename ElemType >
using jeod::JeodList< ElemType >::stl_container_type = std::list<ElemType>
```

The `std::list` itself.

Definition at line 114 of file jeod_list.hh.

8.4.2.4 this_container_type

```
template<typename ElemType >
using jeod::JeodList< ElemType >::this_container_type = JeodList<ElemType>
```

This particular [JeodList](#) type.

Definition at line 99 of file jeod_list.hh.

8.4.3 Constructor & Destructor Documentation

8.4.3.1 ~JeodList()

```
template<typename ElemType >
virtual jeod::JeodList< ElemType >::~~JeodList ( ) [virtual], [default]
```

Destructor.

8.4.3.2 JeodList() [1/3]

```
template<typename ElemType >
jeod::JeodList< ElemType >::JeodList ( ) [protected], [default]
```

Default constructor.

8.4.3.3 JeodList() [2/3]

```
template<typename ElemType >
jeod::JeodList< ElemType >::JeodList (
    const this_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor.

Definition at line 288 of file jeod_list.hh.

8.4.3.4 JeodList() [3/3]

```
template<typename ElemType >
jeod::JeodList< ElemType >::JeodList (
    const stl_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 297 of file jeod_list.hh.

8.4.4 Member Function Documentation

8.4.4.1 merge() [1/2]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::merge (
    stl_container_type & other ) [inline]
```

Merge the contents of some other list into this list, emptying the other list.

Parameters

<i>other</i>	Other list to be merged into this list.
--------------	---

Definition at line 153 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.2 merge() [2/2]

```
template<typename ElemType >
template<typename Compare >
void jeod::JeodList< ElemType >::merge (
    stl_container_type & other,
    Compare comp ) [inline]
```

Merge the contents of some other list into this list using the provided comparator to guide the merge.

The other list is emptied.

Parameters

<i>other</i>	Other list to be merged into this list.
<i>comp</i>	Comparison function.

Definition at line 164 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.3 operator=() [1/2]

```
template<typename ElemType >
JeodList& jeod::JeodList< ElemType >::operator= (
    const stl_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 131 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::operator=().

8.4.4.4 operator=() [2/2]

```
template<typename ElemType >
JeodList& jeod::JeodList< ElemType >::operator= (
    const stl_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 140 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::operator=().

8.4.4.5 pop_front()

```
template<typename ElemType >
void jeod::JeodList< ElemType >::pop_front ( ) [inline]
```

Deletes the element at the head of the list.

Definition at line 181 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.6 push_front()

```
template<typename ElemType >
void jeod::JeodList< ElemType >::push_front (
    const ElemType & elem ) [inline]
```

Add an element to the head of the list.

Parameters

<i>elem</i>	Element to be added.
-------------	----------------------

Definition at line 173 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.7 remove()

```
template<typename ElemType >
void jeod::JeodList< ElemType >::remove (
    const ElemType & value ) [inline]
```

Remove elements from the list that are equal to the provided value.

Definition at line 189 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.8 remove_if()

```
template<typename ElemType >
template<typename Predicate >
void jeod::JeodList< ElemType >::remove_if (
    Predicate pred ) [inline]
```

Remove elements from the list that pass the provided test.

Parameters

<i>pred</i>	Predicate function, which must be able to take a const ref to ElemType as an argument and must return a bool.
-------------	---

Definition at line 199 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.9 reverse()

```
template<typename ElemType >
void jeod::JeodList< ElemType >::reverse ( ) [inline]
```

Reverse the list.

Definition at line 207 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.10 sort() [1/2]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::sort ( ) [inline]
```

Sort using the default comparison operator.

Definition at line 246 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.11 `sort()` [2/2]

```
template<typename ElemType >
template<typename Compare >
void jeod::JeodList< ElemType >::sort (
    Compare comp ) [inline]
```

Sort using the provided comparator.

Parameters

<i>comp</i>	Comparison function, which must be able to take a pair of ElemType as arguments and must return a bool.
-------------	---

Definition at line 256 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.12 `splice()` [1/3]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::splice (
    typename jeod_stl_container_type::iterator position,
    stl_container_type & other ) [inline]
```

Inserts the contents of *other* before *position*, emptying *other*.

Definition at line 215 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.13 `splice()` [2/3]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::splice (
    typename jeod_stl_container_type::iterator position,
    stl_container_type & other,
    typename jeod_stl_container_type::iterator other_pos ) [inline]
```

Inserts the element *other_pos* of *other* before *position*, deleting that element from *other*.

Definition at line 224 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.14 splice() [3/3]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::splice (
    typename jeod_stl_container_type::iterator position,
    stl_container_type & other,
    typename jeod_stl_container_type::iterator first,
    typename jeod_stl_container_type::iterator last ) [inline]
```

Inserts elements in *other* from *first* up to but not including *last* before *position*, deleting those element from *other*.

Definition at line 235 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.15 unique() [1/2]

```
template<typename ElemType >
void jeod::JeodList< ElemType >::unique ( ) [inline]
```

Remove duplicates using the default equality operator.

Definition at line 264 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

8.4.4.16 unique() [2/2]

```
template<typename ElemType >
template<typename BinaryPredicate >
void jeod::JeodList< ElemType >::unique (
    BinaryPredicate comp ) [inline]
```

Remove duplicates using the provided comparator.

Parameters

<i>comp</i>	Comparison function, which must be able to take a pair of ElemType as arguments and must return a bool.
-------------	---

Definition at line 274 of file jeod_list.hh.

References jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::contents.

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

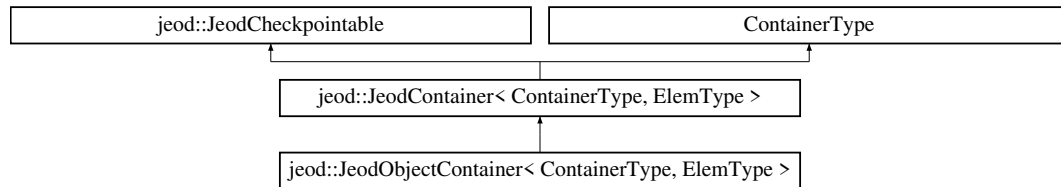
- [jeod_list.hh](#)

8.5 jeod::JeodObjectContainer< ContainerType, ElemType > Class Template Reference

A [JeodObjectContainer](#) is a [JeodContainer](#) that contains objects of type ElemType.

```
#include <object_container.hh>
```

Inheritance diagram for jeod::JeodObjectContainer< ContainerType, ElemType >:



Public Member Functions

- [JeodObjectContainer](#) ()
Construct a [JeodObjectContainer](#).
- [JeodObjectContainer](#) (const [JeodObjectContainer](#) &source)
Copy-construct a [JeodObjectContainer](#).
- [JeodObjectContainer](#) (const typename ContainerType::stl_container_type &source)
Copy-construct a [JeodObjectContainer](#).
- [JeodObjectContainer](#) & operator= (const [JeodObjectContainer](#) &source)
Copy from a [JeodObjectContainer](#).
- [JeodObjectContainer](#) & operator= (const typename ContainerType::stl_container_type &source)
Copy from an STL container.
- virtual [~JeodObjectContainer](#) ()
Destruct a [JeodObjectContainer](#).
- void [pre_checkpoint](#) () override
Prepare to checkpoint a [JeodObjectContainer](#).
- void [post_checkpoint](#) () override
Cleanup after performing a checkpoint.
- void [post_restart](#) () override
Cleanup after performing a restart.
- void [start_checkpoint](#) () override
Prepare to checkpoint the object in question.
- void [advance_checkpoint](#) () override
Advance to the next item to be checkpointed.
- const std::string [get_item_value](#) () override
Return the value of the item to be written to the checkpoint file.
- void [perform_insert_action](#) (const std::string &value) override
Interpret the provided value and add it to the list.
- const std::string [get_final_value](#) () override
Return the value of the item to be written to the checkpoint file.
- void [perform_cleanup_action](#) (const std::string &value) override
Cleanup detritus created during the restore process.

Protected Attributes

- `size_t index`
Index number into the copy; used during checkpoint process.
- `ElemType * copy`
C-style array copy of the object; used during checkpoint process.

Friends

- class `InputProcessor`
- void `init_attrjeod__JeodObjectContainer()`

Additional Inherited Members

8.5.1 Detailed Description

```
template<typename ContainerType, typename ElemType>
class jeod::JeodObjectContainer< ContainerType, ElemType >
```

A `JeodObjectContainer` is a `JeodContainer` that contains objects of type `ElemType`.

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

8.5.2 Constructor & Destructor Documentation

8.5.2.1 `JeodObjectContainer()` [1/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodObjectContainer< ContainerType, ElemType >::JeodObjectContainer ( ) [inline]
```

Construct a `JeodObjectContainer`.

Definition at line 89 of file `object_container.hh`.

8.5.2.2 `JeodObjectContainer()` [2/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodObjectContainer< ContainerType, ElemType >::JeodObjectContainer (
    const JeodObjectContainer< ContainerType, ElemType > & source ) [inline]
```

Copy-construct a `JeodObjectContainer`.

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Object container to be copied.
---------------	--------------------------------

Definition at line 101 of file object_container.hh.

8.5.2.3 JeodObjectContainer() [3/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodObjectContainer< ContainerType, ElemType >::JeodObjectContainer (
    const typename ContainerType::stl_container_type & source ) [inline], [explicit]
```

Copy-construct a [JeodObjectContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Object container to be copied.
---------------	--------------------------------

Definition at line 114 of file object_container.hh.

8.5.2.4 ~JeodObjectContainer()

```
template<typename ContainerType, typename ElemType>
virtual jeod::JeodObjectContainer< ContainerType, ElemType >::~~JeodObjectContainer ( ) [inline],
[virtual]
```

Destruct a [JeodObjectContainer](#).

Definition at line 148 of file object_container.hh.

8.5.3 Member Function Documentation

8.5.3.1 advance_checkpoint()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::advance_checkpoint ( ) [inline],
[override], [virtual]
```

Advance to the next item to be checkpointed.

The local checkpoint index is advanced to keep in sync with the parent class' checkpoint iterator.

Reimplemented from [jeod::JeodContainer< ContainerType, ElemType >](#).

Definition at line 211 of file object_container.hh.

8.5.3.2 `get_final_value()`

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodObjectContainer< ContainerType, ElemType >::get_final_value ( )
[inline], [override], [virtual]
```

Return the value of the item to be written to the checkpoint file.

For a [JeodObjectContainer](#), the value is the name of the corresponding object in the C-style copy of the object's contents.

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

Definition at line 246 of file `object_container.hh`.

8.5.3.3 `get_item_value()`

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodObjectContainer< ContainerType, ElemType >::get_item_value ( )
[inline], [override], [virtual]
```

Return the value of the item to be written to the checkpoint file.

For a [JeodObjectContainer](#), the value is the name of the corresponding object in the C-style copy of the object's contents.

Implements [jeod::JeodCheckpointable](#).

Definition at line 222 of file `object_container.hh`.

8.5.3.4 `operator=()` [1/2]

```
template<typename ContainerType, typename ElemType>
JeodObjectContainer& jeod::JeodObjectContainer< ContainerType, ElemType >::operator= (
    const JeodObjectContainer< ContainerType, ElemType > & source ) [inline]
```

Copy from a [JeodObjectContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Object container to be copied.
---------------	--------------------------------

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

8.5.3.5 operator=() [2/2]

```
template<typename ContainerType, typename ElemType>
JeodObjectContainer& jeod::JeodObjectContainer< ContainerType, ElemType >::operator= (
    const typename ContainerType::stl_container_type & source ) [inline]
```

Copy from an STL container.

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Object container to be copied.
---------------	--------------------------------

Definition at line 139 of file object_container.hh.

8.5.3.6 perform_cleanup_action()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::perform_cleanup_action (
    const std::string & value ) [inline], [override], [virtual]
```

Cleanup detritus created during the restore process.

Here we delete the temporary array created during checkpoint.

Parameters

<i>value</i>	String name of cleanup target.
--------------	--------------------------------

Reimplemented from [jeod::JeodContainer< ContainerType, ElemType >](#).

Definition at line 256 of file object_container.hh.

8.5.3.7 perform_insert_action()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::perform_insert_action (
    const std::string & value ) [inline], [override], [virtual]
```

Interpret the provided value and add it to the list.

For a [JeodObjectContainer](#), the value should name an element of the C-style copy of the object's contents.

Implements [jeod::JeodContainer](#)< ContainerType, ElemType >.

Definition at line 233 of file object_container.hh.

8.5.3.8 post_checkpoint()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::post_checkpoint ( ) [inline],
[override], [virtual]
```

Cleanup after performing a checkpoint.

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

Definition at line 178 of file object_container.hh.

8.5.3.9 post_restart()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::post_restart ( ) [inline], [override],
[virtual]
```

Cleanup after performing a restart.

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

Definition at line 190 of file object_container.hh.

8.5.3.10 pre_checkpoint()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::pre_checkpoint ( ) [inline], [override],
[virtual]
```

Prepare to checkpoint a [JeodObjectContainer](#).

The contents of an object container is checkpointed by allocating a C-style array of the same size as the container and populating the array with copies of the container contents. The existing checkpoint capabilities will checkpoint this array, so all that remains to be done is to associate the array elements with the container.

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

Definition at line 161 of file object_container.hh.

8.5.3.11 start_checkpoint()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodObjectContainer< ContainerType, ElemType >::start_checkpoint ( ) [inline],
[override], [virtual]
```

Prepare to checkpoint the object in question.

The local checkpoint index is initialized to zero to reflect that the parent class' checkpoint iterator starts at the zeroth element.

Reimplemented from [jeod::JeodContainer< ContainerType, ElemType >](#).

Definition at line 200 of file object_container.hh.

8.5.4 Friends And Related Function Documentation

8.5.4.1 init_attrjeod__JeodObjectContainer

```
template<typename ContainerType, typename ElemType>
void init_attrjeod__JeodObjectContainer ( ) [friend]
```

8.5.4.2 InputProcessor

```
template<typename ContainerType, typename ElemType>
friend class InputProcessor [friend]
```

Definition at line 85 of file object_container.hh.

8.5.5 Field Documentation

8.5.5.1 copy

```
template<typename ContainerType, typename ElemType>
ElemType* jeod::JeodObjectContainer< ContainerType, ElemType >::copy [protected]
```

C-style array copy of the object; used during checkpoint process.

```
trick_io(**)
```

Definition at line 274 of file object_container.hh.

8.5.5.2 index

```
template<typename ContainerType, typename ElemType>
size_t jeod::JeodObjectContainer< ContainerType, ElemType >::index [protected]
```

Index number into the copy; used during checkpoint process.

trick_io(**)

Definition at line 269 of file object_container.hh.

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

- [object_container.hh](#)

8.6 jeod::JeodObjectList< ElemType > Class Template Reference

Defines a registry for defining a checkpointable list of objects.

```
#include <object_list.hh>
```

Public Types

- using `type` = [JeodObjectContainer](#)< [JeodList](#)< ElemType >, ElemType >
Template typedef for a checkpointable list of objects.

8.6.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodObjectList< ElemType >
```

Defines a registry for defining a checkpointable list of objects.

Usage: [JeodObjectList](#)<`type`>::`type` variable_name

Definition at line 76 of file object_list.hh.

8.6.2 Member Typedef Documentation

8.6.2.1 type

```
template<typename ElemType >
using jeod::JeodObjectList< ElemType >::type = JeodObjectContainer<JeodList<ElemType>, ElemType>
```

Template typedef for a checkpointable list of objects.

Definition at line 82 of file object_list.hh.

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

- [object_list.hh](#)

8.7 jeod::JeodObjectSet< ElemType > Class Template Reference

Defines a registry for defining a checkpointable set of objects.

```
#include <object_set.hh>
```

Public Types

- using `type` = `JeodObjectContainer< JeodSet< ElemType >, ElemType >`
Template typedef for a checkpointable set of objects.

8.7.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodObjectSet< ElemType >
```

Defines a registry for defining a checkpointable set of objects.

Usage: `JeodObjectSet<type>::type variable_name`

Definition at line 76 of file object_set.hh.

8.7.2 Member Typedef Documentation

8.7.2.1 type

```
template<typename ElemType >
using jeod::JeodObjectSet< ElemType >::type = JeodObjectContainer<JeodSet<ElemType>, ElemType>
```

Template typedef for a checkpointable set of objects.

Definition at line 82 of file object_set.hh.

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

- [object_set.hh](#)

8.8 jeod::JeodObjectVector< ElemType > Class Template Reference

Defines a registry for defining a checkpointable vector of objects.

```
#include <object_vector.hh>
```

Public Types

- using `type` = `JeodObjectContainer`< `JeodVector`< ElemType >, ElemType >
Template typedef for a checkpointable vector of objects.

8.8.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodObjectVector< ElemType >
```

Defines a registry for defining a checkpointable vector of objects.

Usage: `JeodObjectVector<type>::type` variable_name

Definition at line 76 of file `object_vector.hh`.

8.8.2 Member Typedef Documentation

8.8.2.1 type

```
template<typename ElemType >
using jeod::JeodObjectVector< ElemType >::type = JeodObjectContainer<JeodVector<ElemType>,
ElemType>
```

Template typedef for a checkpointable vector of objects.

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

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

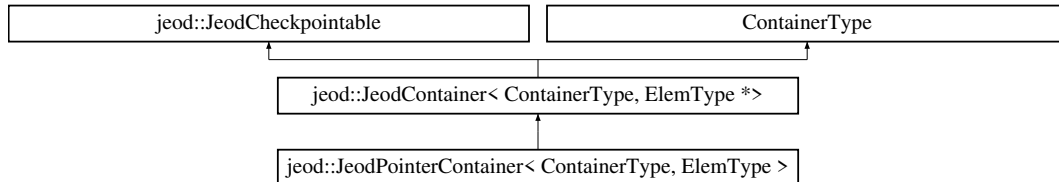
- [object_vector.hh](#)

8.9 jeod::JeodPointerContainer< ContainerType, ElemType > Class Template Reference

A [JeodPointerContainer](#) is a [JeodContainer](#) that contains pointers to objects of type ElemType.

```
#include <pointer_container.hh>
```

Inheritance diagram for jeod::JeodPointerContainer< ContainerType, ElemType >:



Public Member Functions

- [JeodPointerContainer](#) ()
Construct a [JeodPointerContainer](#).
- [JeodPointerContainer](#) (const [JeodPointerContainer](#) &source)
Copy-construct a [JeodPointerContainer](#).
- [JeodPointerContainer](#) (const typename ContainerType::stl_container_type &source)
Copy-construct a [JeodPointerContainer](#).
- [JeodPointerContainer](#) & operator= (const [JeodPointerContainer](#) &source)
Copy from a [JeodPointerContainer](#).
- [JeodPointerContainer](#) & operator= (const typename ContainerType::stl_container_type &source)
Copy from an STL container.
- virtual [~JeodPointerContainer](#) ()=default
Destruct a [JeodPointerContainer](#).
- void [initialize_checkpointable](#) (const void *container, const std::type_info &container_type, const std::string &elem_name) override
Initialize a checkpointable object, called by the checkpoint manager.
- const std::string [get_item_value](#) () override
Return the value of the item to be written to the checkpoint file.
- void [perform_insert_action](#) (const std::string &value) override
Interpret the provided value and add it to the list.

Protected Attributes

- const JeodMemoryTypeDescriptor * [base_type_descriptor](#) {}
Memory model descriptor of the type of data stored in the container.

Additional Inherited Members

8.9.1 Detailed Description

```
template<typename ContainerType, typename ElemType>
class jeod::JeodPointerContainer< ContainerType, ElemType >
```

A [JeodPointerContainer](#) is a [JeodContainer](#) that contains pointers to objects of type ElemType.

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

8.9.2 Constructor & Destructor Documentation

8.9.2.1 JeodPointerContainer() [1/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPointerContainer< ContainerType, ElemType >::JeodPointerContainer ( ) [inline]
```

Construct a [JeodPointerContainer](#).

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

8.9.2.2 JeodPointerContainer() [2/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPointerContainer< ContainerType, ElemType >::JeodPointerContainer (
    const JeodPointerContainer< ContainerType, ElemType > & source ) [inline]
```

Copy-construct a [JeodPointerContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Pointer container to be copied.
---------------	---------------------------------

Definition at line 98 of file `pointer_container.hh`.

8.9.2.3 JeodPointerContainer() [3/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPointerContainer< ContainerType, ElemType >::JeodPointerContainer (
    const typename ContainerType::stl_container_type & source ) [inline], [explicit]
```

Copy-construct a [JeodPointerContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Pointer container to be copied.
---------------	---------------------------------

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

8.9.2.4 ~JeodPointerContainer()

```
template<typename ContainerType, typename ElemType>
virtual jeod::JeodPointerContainer< ContainerType, ElemType >::~~JeodPointerContainer ( )
[virtual], [default]
```

Destruct a [JeodPointerContainer](#).

8.9.3 Member Function Documentation

8.9.3.1 get_item_value()

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodPointerContainer< ContainerType, ElemType >::get_item_value ( )
[inline], [override], [virtual]
```

Return the value of the item to be written to the checkpoint file.

For a [JeodPointerContainer](#), the value names the pointed-to object.

Implements [jeod::JeodCheckpointable](#).

Definition at line 168 of file `pointer_container.hh`.

References [jeod::JeodPointerContainer< ContainerType, ElemType >::base_type_descriptor](#), and [jeod::JeodContainer< ContainerType, ElemType * >::checkpoint_iter](#).

8.9.3.2 initialize_checkpointable()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodPointerContainer< ContainerType, ElemType >::initialize_checkpointable (
    const void * container,
    const std::type_info & container_type,
    const std::string & elem_name ) [inline], [override], [virtual]
```

Initialize a checkpointable object, called by the checkpoint manager.

In the case of a [JeodPointerContainer](#), this method gets the descriptor for the type of data pointed to members of the container.

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

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

References [jeod::JeodPointerContainer< ContainerType, ElemType >::base_type_descriptor](#), and [jeod::JeodContainer< ContainerType, ElemType >::initialize_checkpointable\(\)](#).

8.9.3.3 `operator=()` [1/2]

```
template<typename ContainerType, typename ElemType>
JeodPointerContainer& jeod::JeodPointerContainer< ContainerType, ElemType >::operator= (
    const JeodPointerContainer< ContainerType, ElemType > & source ) [inline]
```

Copy from a [JeodPointerContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Pointer container to be copied.
---------------	---------------------------------

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

References `jeod::JeodContainer< ContainerType, ElemType >::operator=()`.

8.9.3.4 `operator=()` [2/2]

```
template<typename ContainerType, typename ElemType>
JeodPointerContainer& jeod::JeodPointerContainer< ContainerType, ElemType >::operator= (
    const typename ContainerType::stl_container_type & source ) [inline]
```

Copy from an STL container.

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Pointer container to be copied.
---------------	---------------------------------

Definition at line 134 of file `pointer_container.hh`.

References `jeod::JeodContainer< ContainerType, ElemType >::operator=()`.

8.9.3.5 `perform_insert_action()`

```
template<typename ContainerType, typename ElemType>
void jeod::JeodPointerContainer< ContainerType, ElemType >::perform_insert_action (
    const std::string & value ) [inline], [override], [virtual]
```

Interpret the provided value and add it to the list.

For a [JeodPointerContainer](#), the value should specify (in string form) the address of some object in active memory.

Implements [jeod::JeodContainer< ContainerType, ElemType *>](#).

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

8.9.4 Field Documentation

8.9.4.1 base_type_descriptor

```
template<typename ContainerType, typename ElemType>
const JeodMemoryTypeDescriptor* jeod::JeodPointerContainer< ContainerType, ElemType >::base_↵
type_descriptor {} [protected]
```

Memory model descriptor of the type of data stored in the container.

`trick_io(**)`

Definition at line 188 of file `pointer_container.hh`.

Referenced by [jeod::JeodPointerContainer< ContainerType, ElemType >::get_item_value\(\)](#), and [jeod::Jeod↵
PointerContainer< ContainerType, ElemType >::initialize_checkpointable\(\)](#).

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

- [pointer_container.hh](#)

8.10 jeod::JeodPointerList< ElemType > Class Template Reference

Defines a registry for defining a checkpointable list of pointers.

```
#include <pointer_list.hh>
```

Public Types

- using `type = JeodPointerContainer< JeodList< ElemType * >, ElemType >`
Template typedef for a checkpointable list of pointers.

8.10.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPointerList< ElemType >
```

Defines a registry for defining a checkpointable list of pointers.

Usage: [JeodPointerList<type>::type](#) `variable_name`

Definition at line 76 of file `pointer_list.hh`.

8.10.2 Member Typedef Documentation

8.10.2.1 type

```
template<typename ElemType >
using jeod::JeodPointerList< ElemType >::type = JeodPointerContainer<JeodList<ElemType *>,
ElemType>
```

Template typedef for a checkpointable list of pointers.

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

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

- [pointer_list.hh](#)

8.11 jeod::JeodPointerSet< ElemType > Class Template Reference

Defines a registry for defining a checkpointable set of pointers.

```
#include <pointer_set.hh>
```

Public Types

- using `type` = `JeodPointerContainer< JeodSet< ElemType * >, ElemType >`
Template typedef for a checkpointable set of pointers.

8.11.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPointerSet< ElemType >
```

Defines a registry for defining a checkpointable set of pointers.

Usage: `JeodPointerSet<type>::type` `variable_name`

Definition at line 76 of file `pointer_set.hh`.

8.11.2 Member Typedef Documentation

8.11.2.1 type

```
template<typename ElemType >
using jeod::JeodPointerSet< ElemType >::type = JeodPointerContainer<JeodSet<ElemType *>,
ElemType>
```

Template typedef for a checkpointable set of pointers.

Definition at line 82 of file pointer_set.hh.

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

- [pointer_set.hh](#)

8.12 jeod::JeodPointerVector< ElemType > Class Template Reference

Defines a registry for defining a checkpointable vector of pointers.

```
#include <pointer_vector.hh>
```

Public Types

- using `type` = `JeodPointerContainer< JeodVector< ElemType * >, ElemType >`
Template typedef for a checkpointable vector of pointers.

8.12.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPointerVector< ElemType >
```

Defines a registry for defining a checkpointable vector of pointers.

Usage: `JeodPointerVector<type>::type` variable_name

Definition at line 76 of file pointer_vector.hh.

8.12.2 Member Typedef Documentation

8.12.2.1 type

```
template<typename ElemType >
using jeod::JeodPointerVector< ElemType >::type = JeodPointerContainer<JeodVector<ElemType
*>, ElemType>
```

Template typedef for a checkpointable vector of pointers.

Definition at line 82 of file pointer_vector.hh.

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

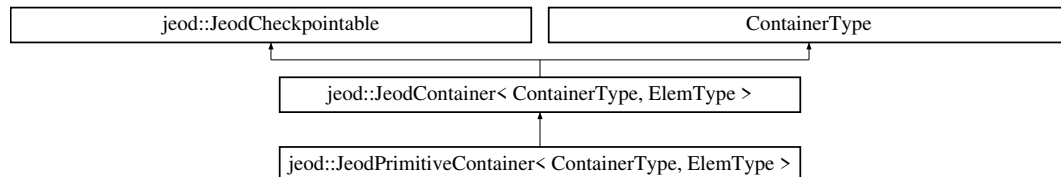
- [pointer_vector.hh](#)

8.13 jeod::JeodPrimitiveContainer< ContainerType, ElemType > Class Template Reference

A [JeodPrimitiveContainer](#) is a [JeodContainer](#) that contains primitive data of type ElemType.

```
#include <primitive_container.hh>
```

Inheritance diagram for jeod::JeodPrimitiveContainer< ContainerType, ElemType >:



Public Member Functions

- [JeodPrimitiveContainer](#) ()=default
Construct a [JeodPrimitiveContainer](#).
- [JeodPrimitiveContainer](#) (const [JeodPrimitiveContainer](#) &source)
Copy-construct a [JeodPrimitiveContainer](#).
- [JeodPrimitiveContainer](#) (const typename ContainerType::stl_container_type &source)
Copy-construct a [JeodPrimitiveContainer](#).
- [JeodPrimitiveContainer](#) & operator= (const [JeodPrimitiveContainer](#) &source)
Copy from a [JeodPrimitiveContainer](#).
- [JeodPrimitiveContainer](#) & operator= (const typename ContainerType::stl_container_type &source)
Copy from an STL container.
- virtual [~JeodPrimitiveContainer](#) ()=default
Destruct a [JeodPrimitiveContainer](#).
- const std::string [get_item_value](#) () override
Return the value of the item to be written to the checkpoint file.
- void [perform_insert_action](#) (const std::string &value) override
Interpret the provided value and insert it at the end of the object.

Protected Attributes

- [JeodPrimitiveSerializer](#)< ElemType > [serializer](#)
Serializer / deserializer.

Additional Inherited Members

8.13.1 Detailed Description

```
template<typename ContainerType, typename ElemType>
class jeod::JeodPrimitiveContainer< ContainerType, ElemType >
```

A [JeodPrimitiveContainer](#) is a [JeodContainer](#) that contains primitive data of type ElemType.

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

8.13.2 Constructor & Destructor Documentation

8.13.2.1 JeodPrimitiveContainer() [1/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPrimitiveContainer< ContainerType, ElemType >::JeodPrimitiveContainer ( ) [default]
```

Construct a [JeodPrimitiveContainer](#).

8.13.2.2 JeodPrimitiveContainer() [2/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPrimitiveContainer< ContainerType, ElemType >::JeodPrimitiveContainer (
    const JeodPrimitiveContainer< ContainerType, ElemType > & source ) [inline]
```

Copy-construct a [JeodPrimitiveContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Primitive container to be copied.
---------------	-----------------------------------

Definition at line 94 of file primitive_container.hh.

8.13.2.3 JeodPrimitiveContainer() [3/3]

```
template<typename ContainerType, typename ElemType>
jeod::JeodPrimitiveContainer< ContainerType, ElemType >::JeodPrimitiveContainer (
    const typename ContainerType::stl_container_type & source ) [inline], [explicit]
```

Copy-construct a [JeodPrimitiveContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Primitive container to be copied.
---------------	-----------------------------------

Definition at line 105 of file primitive_container.hh.

8.13.2.4 ~JeodPrimitiveContainer()

```
template<typename ContainerType, typename ElemType>
virtual jeod::JeodPrimitiveContainer< ContainerType, ElemType >::~~JeodPrimitiveContainer ( )
[virtual], [default]
```

Destruct a [JeodPrimitiveContainer](#).

8.13.3 Member Function Documentation

8.13.3.1 get_item_value()

```
template<typename ContainerType, typename ElemType>
const std::string jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value ( )
[inline], [override], [virtual]
```

Return the value of the item to be written to the checkpoint file.

[JeodPrimitiveContainer](#) use the serializer to translate values to strings.

Implements [jeod::JeodCheckpointable](#).

Definition at line 143 of file primitive_container.hh.

References [jeod::JeodContainer< ContainerType, ElemType >::checkpoint_iter](#), [jeod::JeodPrimitiveContainer< ContainerType, ElemType >::serializer](#), and [jeod::JeodPrimitiveSerializer< Type >::to_string\(\)](#).

8.13.3.2 operator=() [1/2]

```
template<typename ContainerType, typename ElemType>
JeodPrimitiveContainer& jeod::JeodPrimitiveContainer< ContainerType, ElemType >::operator= (
    const JeodPrimitiveContainer< ContainerType, ElemType > & source ) [inline]
```

Copy from a [JeodPrimitiveContainer](#).

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Primitive container to be copied.
---------------	-----------------------------------

Definition at line 116 of file primitive_container.hh.

References [jeod::JeodContainer< ContainerType, ElemType >::operator=\(\)](#).

8.13.3.3 operator=() [2/2]

```
template<typename ContainerType, typename ElemType>
JeodPrimitiveContainer& jeod::JeodPrimitiveContainer< ContainerType, ElemType >::operator= (
    const typename ContainerType::stl_container_type & source ) [inline]
```

Copy from an STL container.

Note

This copies the Container contents, but not the Checkpointable contents.

Parameters

<i>source</i>	Primitive container to be copied.
---------------	-----------------------------------

Definition at line 128 of file primitive_container.hh.

References [jeod::JeodContainer< ContainerType, ElemType >::operator=\(\)](#).

8.13.3.4 perform_insert_action()

```
template<typename ContainerType, typename ElemType>
void jeod::JeodPrimitiveContainer< ContainerType, ElemType >::perform_insert_action (
    const std::string & value ) [inline], [override], [virtual]
```

Interpret the provided value and insert it at the end of the object.

[JeodPrimitiveContainer](#) use the serializer to interpret the input value.

Implements [jeod::JeodContainer< ContainerType, ElemType >](#).

Definition at line 152 of file primitive_container.hh.

References [jeod::JeodPrimitiveSerializer< Type >::from_string\(\)](#), and [jeod::JeodPrimitiveContainer< ContainerType, ElemType >::serializer](#).

8.13.4 Field Documentation

8.13.4.1 serializer

```
template<typename ContainerType, typename ElemType>
JeodPrimitiveSerializer<ElemType> jeod::JeodPrimitiveContainer< ContainerType, ElemType >↵
::serializer [protected]
```

Serializer / deserializer.

trick_io(**)

Definition at line 163 of file primitive_container.hh.

Referenced by jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value(), and jeod::Jeod↵
PrimitiveContainer< ContainerType, ElemType >::perform_insert_action().

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

- [primitive_container.hh](#)

8.14 jeod::JeodPrimitiveList< ElemType > Class Template Reference

Defines a registry for defining a checkpointable list of primitives.

```
#include <primitive_list.hh>
```

Public Types

- using [type](#) = [JeodPrimitiveContainer](#)< [JeodList](#)< ElemType >, ElemType >
Template typedef for a checkpointable list of primitives.

8.14.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPrimitiveList< ElemType >
```

Defines a registry for defining a checkpointable list of primitives.

Usage: [JeodPrimitiveList](#)<[type](#)>::[type](#) variable_name

Definition at line 76 of file primitive_list.hh.

8.14.2 Member Typedef Documentation

8.14.2.1 type

```
template<typename ElemType >
using jeod::JeodPrimitiveList< ElemType >::type = JeodPrimitiveContainer<JeodList<ElemType>,
ElemType>
```

Template typedef for a checkpointable list of primitives.

Definition at line 82 of file primitive_list.hh.

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

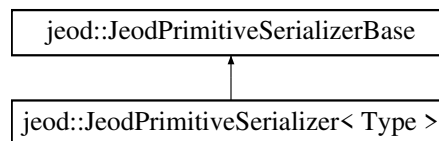
- [primitive_list.hh](#)

8.15 jeod::JeodPrimitiveSerializer< Type > Class Template Reference

Serializer / deserializer for primitive data.

```
#include <primitive_serializer.hh>
```

Inheritance diagram for jeod::JeodPrimitiveSerializer< Type >:



Public Member Functions

- [JeodPrimitiveSerializer](#) ()=default
Construct a [JeodPrimitiveSerializer](#).
- [~JeodPrimitiveSerializer](#) () override=default
Destruct a [JeodPrimitiveSerializer](#).
- const std::string [to_string](#) (const Type &val)
Convert a primitive value to its string-equivalent.
- Type [from_string](#) (const std::string &val)
Convert a string to its corresponding primitive value.
- [JeodPrimitiveSerializer](#) (const [JeodPrimitiveSerializer](#) &)=delete
- [JeodPrimitiveSerializer](#) & operator= (const [JeodPrimitiveSerializer](#) &)=delete
- template<>
const std::string [to_string](#) (const std::string &val)
Convert a string to a string suitable for output: Backslash-escape backslashes and double quotes.
- template<>
std::string [from_string](#) (const std::string &val)
Convert a string as recorded in the checkpoint file to its original form.
- template<>
const std::string [to_string](#) (const float &val)
Convert a float to a string.

- `template<>`
`float from_string (const std::string &val)`
Convert a string to a float.
- `template<>`
`const std::string to_string (const double &val)`
Convert a double to a string.
- `template<>`
`double from_string (const std::string &val)`
Convert a string to a double.
- `template<>`
`const std::string to_string (const long double &val)`
Convert a long double to a string.
- `template<>`
`long double from_string (const std::string &val)`
Convert a string to a long double.

Additional Inherited Members

8.15.1 Detailed Description

```
template<typename Type>
class jeod::JeodPrimitiveSerializer< Type >
```

Serializer / deserializer for primitive data.

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

8.15.2 Constructor & Destructor Documentation

8.15.2.1 JeodPrimitiveSerializer() [1/2]

```
template<typename Type>
jeod::JeodPrimitiveSerializer< Type >::JeodPrimitiveSerializer ( ) [default]
```

Construct a `JeodPrimitiveSerializer`.

8.15.2.2 ~JeodPrimitiveSerializer()

```
template<typename Type>
jeod::JeodPrimitiveSerializer< Type >::~~JeodPrimitiveSerializer ( ) [override], [default]
```

Destruct a `JeodPrimitiveSerializer`.

8.15.2.3 JeodPrimitiveSerializer() [2/2]

```
template<typename Type>
jeod::JeodPrimitiveSerializer< Type >::JeodPrimitiveSerializer (
    const JeodPrimitiveSerializer< Type > & ) [delete]
```

8.15.3 Member Function Documentation**8.15.3.1 from_string()** [1/5]

```
template<typename Type>
Type jeod::JeodPrimitiveSerializer< Type >::from_string (
    const std::string & val ) [inline]
```

Convert a string to its corresponding primitive value.

Definition at line 130 of file primitive_serializer.hh.

Referenced by jeod::JeodPrimitiveContainer< ContainerType, ElemType >::perform_insert_action().

8.15.3.2 from_string() [2/5]

```
template<>
std::string jeod::JeodPrimitiveSerializer< std::string >::from_string (
    const std::string & val ) [inline]
```

Convert a string as recorded in the checkpoint file to its original form.

Definition at line 154 of file primitive_serializer.hh.

8.15.3.3 from_string() [3/5]

```
template<>
float jeod::JeodPrimitiveSerializer< float >::from_string (
    const std::string & val ) [inline]
```

Convert a string to a float.

Definition at line 170 of file primitive_serializer.hh.

8.15.3.4 from_string() [4/5]

```
template<>
double jeod::JeodPrimitiveSerializer< double >::from_string (
    const std::string & val ) [inline]
```

Convert a string to a double.

Definition at line 186 of file primitive_serializer.hh.

8.15.3.5 from_string() [5/5]

```
template<>
long double jeod::JeodPrimitiveSerializer< long double >::from_string (
    const std::string & val ) [inline]
```

Convert a string to a long double.

Definition at line 202 of file primitive_serializer.hh.

8.15.3.6 operator=()

```
template<typename Type>
JeodPrimitiveSerializer& jeod::JeodPrimitiveSerializer< Type >::operator= (
    const JeodPrimitiveSerializer< Type > & ) [delete]
```

8.15.3.7 to_string() [1/5]

```
template<typename Type>
const std::string jeod::JeodPrimitiveSerializer< Type >::to_string (
    const Type & val ) [inline]
```

Convert a primitive value to its string-equivalent.

Definition at line 120 of file primitive_serializer.hh.

Referenced by jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value().

8.15.3.8 to_string() [2/5]

```
template<>
const std::string jeod::JeodPrimitiveSerializer< std::string >::to_string (
    const std::string & val ) [inline]
```

Convert a string to a string suitable for output: Backslash-escape backslashes and double quotes.

Definition at line 146 of file primitive_serializer.hh.

8.15.3.9 to_string() [3/5]

```
template<>
const std::string jeod::JeodPrimitiveSerializer< float >::to_string (
    const float & val ) [inline]
```

Convert a float to a string.

Definition at line 162 of file primitive_serializer.hh.

8.15.3.10 to_string() [4/5]

```
template<>
const std::string jeod::JeodPrimitiveSerializer< double >::to_string (
    const double & val ) [inline]
```

Convert a double to a string.

Definition at line 178 of file primitive_serializer.hh.

8.15.3.11 to_string() [5/5]

```
template<>
const std::string jeod::JeodPrimitiveSerializer< long double >::to_string (
    const long double & val ) [inline]
```

Convert a long double to a string.

Definition at line 194 of file primitive_serializer.hh.

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

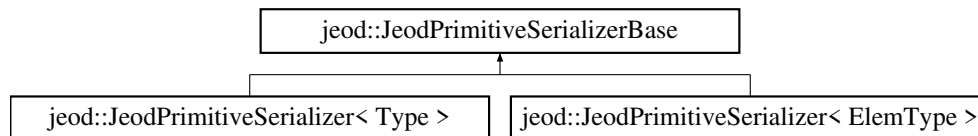
- [primitive_serializer.hh](#)

8.16 jeod::JeodPrimitiveSerializerBase Class Reference

Base class for serializing / deserializing primitive data.

```
#include <primitive_serializer.hh>
```

Inheritance diagram for jeod::JeodPrimitiveSerializerBase:



Public Member Functions

- [JeodPrimitiveSerializerBase](#) ()=default
Construct a [JeodPrimitiveSerializerBase](#).
- virtual [~JeodPrimitiveSerializerBase](#) ()=default
Destruct a [JeodPrimitiveSerializerBase](#).

Static Protected Member Functions

- static const std::string [serialize_string](#) (const std::string &val)
Convert a string to a string suitable for output.
- static const std::string [deserialize_string](#) (const std::string &val)
Convert a serialized string to its internal representation.
- static const std::string [serialize_float](#) (const float &val)
Convert a float to a string suitable for output.
- static float [deserialize_float](#) (const std::string &val)
Convert a serialized float to its internal representation.
- static const std::string [serialize_double](#) (const double &val)
Convert a double to a string suitable for output.
- static double [deserialize_double](#) (const std::string &val)
Convert a serialized double to its internal representation.
- static const std::string [serialize_long_double](#) (const long double &val)
Convert a long double to a string suitable for output.
- static long double [deserialize_long_double](#) (const std::string &val)
Convert a serialized double to its internal representation.

8.16.1 Detailed Description

Base class for serializing / deserializing primitive data.

Definition at line 77 of file primitive_serializer.hh.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 JeodPrimitiveSerializerBase()

```
jeod::JeodPrimitiveSerializerBase::JeodPrimitiveSerializerBase ( ) [default]
```

Construct a [JeodPrimitiveSerializerBase](#).

8.16.2.2 ~JeodPrimitiveSerializerBase()

```
virtual jeod::JeodPrimitiveSerializerBase::~~JeodPrimitiveSerializerBase ( ) [virtual], [default]
```

Destruct a [JeodPrimitiveSerializerBase](#).

8.16.3 Member Function Documentation

8.16.3.1 deserialize_double()

```
double jeod::JeodPrimitiveSerializerBase::deserialize_double (
    const std::string & val ) [static], [protected]
```

Convert a serialized double to its internal representation.

Returns

Deserialized double

Parameters

in	<i>val</i>	Serialized string
----	------------	-------------------

Definition at line 227 of file `primitive_serializer.cc`.

8.16.3.2 deserialize_float()

```
float jeod::JeodPrimitiveSerializerBase::deserialize_float (
    const std::string & val ) [static], [protected]
```

Convert a serialized float to its internal representation.

Returns

Deserialized float

Parameters

in	val	Serialized string
----	-----	-------------------

Definition at line 163 of file primitive_serializer.cc.

8.16.3.3 deserialize_long_double()

```
long double jeod::JeodPrimitiveSerializerBase::deserialize_long_double (
    const std::string & val ) [static], [protected]
```

Convert a serialized double to its internal representation.

Returns

Deserialized long double

Parameters

in	val	Serialized string
----	-----	-------------------

Definition at line 291 of file primitive_serializer.cc.

8.16.3.4 deserialize_string()

```
const std::string jeod::JeodPrimitiveSerializerBase::deserialize_string (
    const std::string & val ) [static], [protected]
```

Convert a serialized string to its internal representation.

Backslash-escaped characters are converted to special characters.

Returns

Deserialized string

Parameters

in	val	Serialized string
----	-----	-------------------

Definition at line 89 of file primitive_serializer.cc.

8.16.3.5 serialize_double()

```
const std::string jeod::JeodPrimitiveSerializerBase::serialize_double (
    const double & val ) [static], [protected]
```

Convert a double to a string suitable for output.

NaNs and Infs get special treatment. Everything is serialized via c++ I/O.

Returns

Serialized number

Parameters

in	val	Number to serialize
----	-----	---------------------

Definition at line 195 of file primitive_serializer.cc.

8.16.3.6 serialize_float()

```
const std::string jeod::JeodPrimitiveSerializerBase::serialize_float (
    const float & val ) [static], [protected]
```

Convert a float to a string suitable for output.

NaNs and Infs get special treatment. Everything is serialized via c++ I/O.

Returns

Serialized number

Parameters

in	val	Number to serialize
----	-----	---------------------

Definition at line 131 of file primitive_serializer.cc.

8.16.3.7 serialize_long_double()

```
const std::string jeod::JeodPrimitiveSerializerBase::serialize_long_double (
    const long double & val ) [static], [protected]
```

Convert a long double to a string suitable for output.

NaNs and Infs get special treatment. Everything is serialized via c++ I/O.

Returns

Serialized number

Parameters

in	val	Number to serialize
----	-----	---------------------

Definition at line 259 of file primitive_serializer.cc.

8.16.3.8 serialize_string()

```
const std::string jeod::JeodPrimitiveSerializerBase::serialize_string (
    const std::string & val ) [static], [protected]
```

Convert a string to a string suitable for output.

Special characters are backslash-escaped.

Returns

Serialized string

Parameters

in	val	String to serialize
----	-----	---------------------

Definition at line 45 of file primitive_serializer.cc.

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

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

8.17 jeod::JeodPrimitiveSet< ElemType > Class Template Reference

Defines a registry for defining a checkpointable set of primitives.

```
#include <primitive_set.hh>
```

Public Types

- using [type](#) = [JeodPrimitiveContainer](#)< [JeodSet](#)< ElemType >, ElemType >
Template typedef for a checkpointable set of primitives.

8.17.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPrimitiveSet< ElemType >
```

Defines a registry for defining a checkpointable set of primitives.

Usage: [JeodPrimitiveSet<type>::type](#) variable_name

Definition at line 75 of file primitive_set.hh.

8.17.2 Member Typedef Documentation

8.17.2.1 type

```
template<typename ElemType >
using jeod::JeodPrimitiveSet< ElemType >::type = JeodPrimitiveContainer<JeodSet<ElemType>,
ElemType>
```

Template typedef for a checkpointable set of primitives.

Definition at line 81 of file primitive_set.hh.

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

- [primitive_set.hh](#)

8.18 jeod::JeodPrimitiveVector< ElemType > Class Template Reference

Defines a registry for defining a checkpointable vector of primitives.

```
#include <primitive_vector.hh>
```

Public Types

- using [type](#) = [JeodPrimitiveContainer](#)< [JeodVector](#)< ElemType >, ElemType >
Template typedef for a checkpointable vector of primitives.

8.18.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodPrimitiveVector< ElemType >
```

Defines a registry for defining a checkpointable vector of primitives.

Usage: [JeodPrimitiveVector<type>::type](#) variable_name

Definition at line 76 of file primitive_vector.hh.

8.18.2 Member Typedef Documentation

8.18.2.1 type

```
template<typename ElemType >
using jeod::JeodPrimitiveVector< ElemType >::type = JeodPrimitiveContainer<JeodVector<ElemType>, ElemType>
```

Template typedef for a checkpointable vector of primitives.

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

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

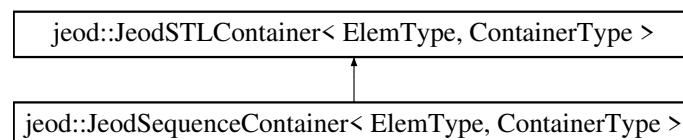
- [primitive_vector.hh](#)

8.19 jeod::JeodSequenceContainer< ElemType, ContainerType > Class Template Reference

This is the base class for the JEOD replacements of the STL sequence containers.

```
#include <jeod_sequence_container.hh>
```

Inheritance diagram for `jeod::JeodSequenceContainer< ElemType, ContainerType >`:



Public Types

- using `this_container_type` = `JeodSequenceContainer< ElemType, ContainerType >`
This type.
- using `base_container_type` = `JeodSTLContainer< ElemType, ContainerType >`
The `JeodSTLContainer`.

Public Member Functions

- virtual [~JeodSequenceContainer](#) ()=default
Destructor.
- [base_container_type::reference back](#) ()
Get the element at the tail of the contents.
- [base_container_type::const_reference back](#) () const
Get the element at the tail of the contents.
- [base_container_type::reference front](#) ()
Get the element at the head of the contents.
- [base_container_type::const_reference front](#) () const
Get the element at the head of the contents.
- template<class InputIterator >
void [assign](#) (InputIterator first, InputIterator last)
Replace the container's contents with that specified by the iterators.
- void [assign](#) (typename [base_container_type::size_type](#) new_size, const ElemType &new_elem)
Replace the container's contents with new_size copies of new_elem.
- [base_container_type::iterator erase](#) (typename [base_container_type::iterator](#) position)
Erase one item.
- [base_container_type::iterator erase](#) (typename [base_container_type::iterator](#) first, typename [base_container_type::iterator](#) last)
Erase a sequence of items.
- template<class InputIterator >
void [insert](#) (typename [base_container_type::iterator](#) position, InputIterator first, InputIterator last)
Insert elements before iterator position, initializing the inserted elements from the values pointed to by an iterator.
- void [insert](#) (typename [base_container_type::iterator](#) position, typename [base_container_type::size_type](#) ncopies, const ElemType &new_elem)
Extends the list by ncopies elements before the iterator position, initializing each newly created element with new_↔elem.
- void [resize](#) (typename [base_container_type::size_type](#) new_size, ElemType new_elem=ElemType())
Resizes the container, adding or deleting items as needed.
- void [push_back](#) (const ElemType &elem)
Add an element to the end of the contents.
- void [pop_back](#) ()
Deletes the element at the end of the contents.
- [iterator insert](#) ([iterator](#) position, const [value_type](#) &new_elem)
Insert a new element initialized with new_elem before the iterator position.

Protected Member Functions

- [JeodSequenceContainer](#) ()=default
Default constructor.
- [JeodSequenceContainer](#) (const [this_container_type](#) &src)
Copy constructor.
- [JeodSequenceContainer](#) (const ContainerType &src)
Copy constructor from STL container.

Additional Inherited Members

8.19.1 Detailed Description

```
template<typename ElemType, typename ContainerType>
class jeod::JeodSequenceContainer< ElemType, ContainerType >
```

This is the base class for the JEOD replacements of the STL sequence containers.

The class derives from [JeodSTLContainer](#), the base class for the JEOD replacements of the STL containers.

A key goal of the JEOD STL sequence container replacement effort is to provide checkpointable replacements that transparently provide the full functionality of the ISO/IEC 14882:2003 STL sequence containers. This class begins that effort by defining types and member functions common to the STL deque, list, and vector class templates. Non-common methods are the responsibility of derived class templates specialized to a specific container types.

Note

Exceptions to full functionality goal: The above goal is not and never will be fully achieved. Exceptions are:

- JEOD doesn't supply a replacement for `std::deque`. JEOD doesn't use deques.
- The full panoply of STL sequence container constructors is not supplied.

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

8.19.2 Member Typedef Documentation

8.19.2.1 base_container_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSequenceContainer< ElemType, ContainerType >::base_container_type = JeodSTLContainer<Elem↵
Type, ContainerType>
```

The [JeodSTLContainer](#).

Definition at line 113 of file `jeod_sequence_container.hh`.

8.19.2.2 this_container_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSequenceContainer< ElemType, ContainerType >::this_container_type = JeodSequenceContainer<Elem↵
Type, ContainerType>
```

This type.

Definition at line 108 of file `jeod_sequence_container.hh`.

8.19.3 Constructor & Destructor Documentation

8.19.3.1 ~JeodSequenceContainer()

```
template<typename ElemType, typename ContainerType>
virtual jeod::JeodSequenceContainer< ElemType, ContainerType >::~~JeodSequenceContainer ( )
[virtual], [default]
```

Destructor.

8.19.3.2 JeodSequenceContainer() [1/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSequenceContainer< ElemType, ContainerType >::JeodSequenceContainer ( ) [protected],
[default]
```

Default constructor.

Note: Making this protected precludes someone from declaring an object to be of type JEODSTLContainer. Access is via some other class that inherits from this class.

8.19.3.3 JeodSequenceContainer() [2/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSequenceContainer< ElemType, ContainerType >::JeodSequenceContainer (
    const this_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor.

Parameters

<code>src</code>	Source container to be copied
------------------	-------------------------------

Definition at line 281 of file jeod_sequence_container.hh.

8.19.3.4 JeodSequenceContainer() [3/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSequenceContainer< ElemType, ContainerType >::JeodSequenceContainer (
    const ContainerType & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 290 of file jeod_sequence_container.hh.

8.19.4 Member Function Documentation

8.19.4.1 assign() [1/2]

```
template<typename ElemType, typename ContainerType>
template<class InputIterator >
void jeod::JeodSequenceContainer< ElemType, ContainerType >::assign (
    InputIterator first,
    InputIterator last ) [inline]
```

Replace the container's contents with that specified by the iterators.

Parameters

<i>first</i>	Input iterator.
<i>last</i>	Input iterator.

Definition at line 171 of file jeod_sequence_container.hh.

8.19.4.2 assign() [2/2]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSequenceContainer< ElemType, ContainerType >::assign (
    typename base_container_type::size_type new_size,
    const ElemType & new_elem ) [inline]
```

Replace the container's contents with *new_size* copies of *new_elem*.

Parameters

<i>new_size</i>	New size of the container.
<i>new_elem</i>	Element to be replicated to fill the container.

Definition at line 181 of file jeod_sequence_container.hh.

8.19.4.3 back() [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::reference jeod::JeodSequenceContainer< ElemType, ContainerType >::back (
) [inline]
```

Get the element at the tail of the contents.

Definition at line 135 of file jeod_sequence_container.hh.

8.19.4.4 back() [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::const_reference jeod::JeodSequenceContainer< ElemType, ContainerType >↵
::back ( ) const [inline]
```

Get the element at the tail of the contents.

Definition at line 143 of file jeod_sequence_container.hh.

8.19.4.5 erase() [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::iterator jeod::JeodSequenceContainer< ElemType, ContainerType >::erase (
    typename base_container_type::iterator position ) [inline]
```

Erase one item.

Parameters

<i>position</i>	Position to be erased
-----------------	-----------------------

Definition at line 190 of file jeod_sequence_container.hh.

8.19.4.6 erase() [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::iterator jeod::JeodSequenceContainer< ElemType, ContainerType >::erase (
    typename base_container_type::iterator first,
    typename base_container_type::iterator last ) [inline]
```

Erase a sequence of items.

Parameters

<i>first</i>	First element to be erased
<i>last</i>	One past last element to be erased

Definition at line 200 of file jeod_sequence_container.hh.

8.19.4.7 front() [1/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::reference jeod::JeodSequenceContainer< ElemType, ContainerType >::front (
) [inline]
```

Get the element at the head of the contents.

Definition at line 151 of file jeod_sequence_container.hh.

8.19.4.8 front() [2/2]

```
template<typename ElemType, typename ContainerType>
base_container_type::const_reference jeod::JeodSequenceContainer< ElemType, ContainerType >↔
::front ( ) const [inline]
```

Get the element at the head of the contents.

Definition at line 159 of file jeod_sequence_container.hh.

8.19.4.9 insert() [1/3]

```
template<typename ElemType, typename ContainerType>
iterator jeod::JeodSTLContainer< ElemType, ContainerType >::insert [inline]
```

Insert a new element initialized with *new_elem* before the iterator *position*.

Parameters

<i>position</i>	Insertion position
<i>new_elem</i>	Element value to be inserted

Returns

Iterator that points to the newly-inserted element

Definition at line 340 of file jeod_stl_container.hh.

8.19.4.10 insert() [2/3]

```
template<typename ElemType, typename ContainerType>
template<class InputIterator >
void jeod::JeodSequenceContainer< ElemType, ContainerType >::insert (
    typename base_container_type::iterator position,
    InputIterator first,
    InputIterator last ) [inline]
```

Insert elements before iterator *position*, initializing the inserted elements from the values pointed to by an iterator.

Parameters

<i>position</i>	Insertion position
<i>first</i>	Input iterator
<i>last</i>	Input iterator

Definition at line 218 of file jeod_sequence_container.hh.

8.19.4.11 insert() [3/3]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSequenceContainer< ElemType, ContainerType >::insert (
    typename base_container_type::iterator position,
    typename base_container_type::size_type ncopies,
    const ElemType & new_elem ) [inline]
```

Extends the list by *ncopies* elements before the iterator *position*, initializing each newly created element with *new_elem*.

Parameters

<i>position</i>	Insertion position
<i>ncopies</i>	Number of elements to be inserted
<i>new_elem</i>	Element value to be inserted

Definition at line 230 of file jeod_sequence_container.hh.

8.19.4.12 pop_back()

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSequenceContainer< ElemType, ContainerType >::pop_back ( ) [inline]
```

Deletes the element at the end of the contents.

Definition at line 259 of file jeod_sequence_container.hh.

8.19.4.13 push_back()

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSequenceContainer< ElemType, ContainerType >::push_back (
    const ElemType & elem ) [inline]
```

Add an element to the end of the contents.

Parameters

<i>elem</i>	Element to be added.
-------------	----------------------

Definition at line 251 of file jeod_sequence_container.hh.

8.19.4.14 resize()

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSequenceContainer< ElemType, ContainerType >::resize (
    typename base_container_type::size_type new_size,
    ElemType new_elem = ElemType() ) [inline]
```

Resizes the container, adding or deleting items as needed.

Parameters

<i>new_size</i>	New size
<i>new_elem</i>	Element to be added repetively if object is to grow.

Definition at line 242 of file jeod_sequence_container.hh.

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

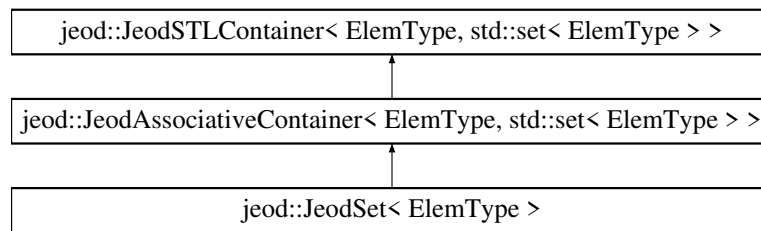
- [jeod_sequence_container.hh](#)

8.20 jeod::JeodSet< ElemType > Class Template Reference

The JEOD replacement for std::set.

```
#include <jeod_set.hh>
```

Inheritance diagram for jeod::JeodSet< ElemType >:



Public Types

- using `this_container_type` = `JeodSet< ElemType >`
This particular JeodSet type.
- using `jeod_associative_container_type` = `JeodAssociativeContainer< ElemType, std::set< ElemType > >`
The JeodAssociativeContainer type.
- using `jeod_stl_container_type` = `JeodSTLContainer< ElemType, std::set< ElemType > >`
The JeodSTLContainer type.
- using `stl_container_type` = `std::set< ElemType >`
The std::set itself.

Public Member Functions

- virtual `~JeodSet()`=default
Destructor.
- `JeodSet & operator=` (const `this_container_type` &src)
Copy contents from the given source.
- `JeodSet & operator=` (const `stl_container_type` &src)
Copy contents from the given source.

Protected Member Functions

- `JeodSet()`=default
Default constructor.
- `JeodSet` (const `this_container_type` &src)
Copy constructor.
- `JeodSet` (const `stl_container_type` &src)
Copy constructor from STL container.

Additional Inherited Members

8.20.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodSet< ElemType >
```

The JEOD replacement for `std::set`.

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

8.20.2 Member Typedef Documentation

8.20.2.1 jeod_associative_container_type

```
template<typename ElemType >  
using jeod::JeodSet< ElemType >::jeod_associative_container_type = JeodAssociativeContainer<ElemType,  
std::set<ElemType> >
```

The [JeodAssociativeContainer](#) type.

Definition at line 94 of file jeod_set.hh.

8.20.2.2 jeod_stl_container_type

```
template<typename ElemType >  
using jeod::JeodSet< ElemType >::jeod_stl_container_type = JeodSTLContainer<ElemType, std::set<ElemType> >
```

The [JeodSTLContainer](#) type.

Definition at line 99 of file jeod_set.hh.

8.20.2.3 stl_container_type

```
template<typename ElemType >  
using jeod::JeodSet< ElemType >::stl_container_type = std::set<ElemType>
```

The `std::set` itself.

Definition at line 104 of file jeod_set.hh.

8.20.2.4 this_container_type

```
template<typename ElemType >  
using jeod::JeodSet< ElemType >::this_container_type = JeodSet<ElemType>
```

This particular [JeodSet](#) type.

Definition at line 89 of file jeod_set.hh.

8.20.3 Constructor & Destructor Documentation

8.20.3.1 ~JeodSet()

```
template<typename ElemType >
virtual jeod::JeodSet< ElemType >::~~JeodSet ( ) [virtual], [default]
```

Destructor.

8.20.3.2 JeodSet() [1/3]

```
template<typename ElemType >
jeod::JeodSet< ElemType >::JeodSet ( ) [protected], [default]
```

Default constructor.

8.20.3.3 JeodSet() [2/3]

```
template<typename ElemType >
jeod::JeodSet< ElemType >::JeodSet (
    const this_container_type & src ) [inline], [protected]
```

Copy constructor.

Definition at line 145 of file jeod_set.hh.

8.20.3.4 JeodSet() [3/3]

```
template<typename ElemType >
jeod::JeodSet< ElemType >::JeodSet (
    const stl_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

src	Source container to be copied
-----	-------------------------------

Definition at line 154 of file jeod_set.hh.

8.20.4 Member Function Documentation

8.20.4.1 operator=() [1/2]

```
template<typename ElemType >
JeodSet& jeod::JeodSet< ElemType >::operator= (
    const this_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 121 of file jeod_set.hh.

References jeod::JeodSTLContainer< ElemType, std::set< ElemType > >::operator=().

8.20.4.2 operator=() [2/2]

```
template<typename ElemType >
JeodSet& jeod::JeodSet< ElemType >::operator= (
    const stl_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 130 of file jeod_set.hh.

References jeod::JeodSTLContainer< ElemType, std::set< ElemType > >::operator=().

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

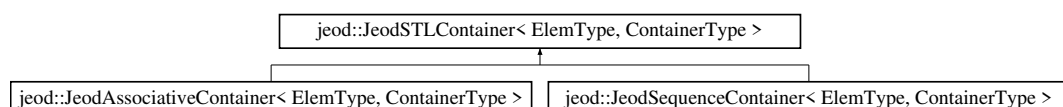
- [jeod_set.hh](#)

8.21 jeod::JeodSTLContainer< ElemType, ContainerType > Class Template Reference

This is the base class for the JEOD replacements of the STL containers.

```
#include <jeod_stl_container.hh>
```

Inheritance diagram for jeod::JeodSTLContainer< ElemType, ContainerType >:



Public Types

- using [this_container_type](#) = JeodSTLContainer< ElemType, ContainerType >
This particular JeodSTLContainer type.
- using [allocator_type](#) = typename ContainerType::allocator_type
Import the ContainerType::allocator_type.
- using [reference](#) = typename ContainerType::reference
Import the ContainerType::reference.
- using [const_reference](#) = typename ContainerType::const_reference
Import the ContainerType::const_reference.
- using [iterator](#) = typename ContainerType::iterator
Import the ContainerType::iterator.
- using [const_iterator](#) = typename ContainerType::const_iterator
Import the ContainerType::const_iterator.
- using [reverse_iterator](#) = typename ContainerType::reverse_iterator
Import the ContainerType::reverse_iterator.
- using [const_reverse_iterator](#) = typename ContainerType::const_reverse_iterator
Import the ContainerType::const_reverse_iterator.
- using [difference_type](#) = typename ContainerType::difference_type
Import the ContainerType::difference_type.
- using [size_type](#) = typename ContainerType::size_type
Import the ContainerType::size_type.
- using [value_type](#) = typename ContainerType::value_type
Import the ContainerType::value_type.

Public Member Functions

- virtual [~JeodSTLContainer](#) ()=default
Destructor.
- [operator ContainerType & \(\)](#)
Returns the contents as an lvalue.
- [operator const ContainerType & \(\) const](#)
Returns the contents as a const rvalue.
- [this_container_type & operator=](#) (const [this_container_type](#) &src)
Assignment operator.
- [this_container_type & operator=](#) (const ContainerType &src)
Assignment operator.
- [allocator_type get_allocator](#) () const
Returns the allocator object used to construct the contents.
- [iterator begin](#) ()
Returns an iterator that points to the first element.
- [const_iterator begin](#) () const
Returns a const iterator that points to the first element.
- [iterator end](#) ()
Returns an iterator that points past the last element.
- [const_iterator end](#) () const
Returns a const iterator that points past the last element.
- [reverse_iterator rbegin](#) ()
Returns a reverse iterator that points to the last element.
- [const_reverse_iterator rbegin](#) () const

- Returns a const reverse iterator that points to the last element.*
- `reverse_iterator rend ()`
Returns a reverse iterator that points before the first element.
- `const_reverse_iterator rend () const`
Returns a const reverse iterator that points before the first element.
- `bool empty () const`
Returns true if the contents are empty, false otherwise.
- `size_type max_size () const`
Returns the implementation's limit on the number of elements.
- `size_type size () const`
Returns the number of elements.
- `void clear ()`
Clear the contents.
- `iterator insert (iterator position, const value_type &new_elem)`
Insert a new element initialized with new_elem before the iterator position.

Protected Member Functions

- `JeodSTLContainer ()=default`
Default constructor.
- `JeodSTLContainer (const this_container_type &src)`
Copy constructor.
- `JeodSTLContainer (const ContainerType &src)`
Copy constructor from STL container.
- `void swap (this_container_type &other)`
Swap contents.
- `void swap (ContainerType &other)`
Swap contents.

Protected Attributes

- `ContainerType contents`
The STL container.

8.21.1 Detailed Description

```
template<typename ElemType, typename ContainerType>
class jeod::JeodSTLContainer< ElemType, ContainerType >
```

This is the base class for the JEOD replacements of the STL containers.

A key goal of the JEOD STL container replacement effort is to provide checkpointable replacements that transparently provide the full functionality of the ISO/IEC 14882:2003 STL containers. This class begins that effort by defining types and member functions common to the STL deque, list, map, set, and vector class templates. Non-common methods are the responsibility of derived class templates specialized to a specific container types.

Note

Exceptions to full functionality goal: The above goal is not and never will be fully achieved. Exceptions are:

- JEOD doesn't supply a replacement for std::deque or std::map. JEOD doesn't use deques at all and its maps are not checkpointable.
- The full panoply of STL container constructors is not supplied.
- The swap method is supplied but it is protected. The intent is that this class be further derived to create a checkpointable class. Swapping the checkpointable content is a dubious concept. The swap method is eventually exposed as the swap_stl_contents method to make it clear that that method is not a true swap.

Definition at line 98 of file jeod_stl_container.hh.

8.21.2 Member Typedef Documentation**8.21.2.1 allocator_type**

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::allocator_type = typename ContainerType::allocator_type
```

Import the ContainerType::allocator_type.

Definition at line 111 of file jeod_stl_container.hh.

8.21.2.2 const_iterator

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::const_iterator = typename ContainerType::const_iterator
```

Import the ContainerType::const_iterator.

Definition at line 131 of file jeod_stl_container.hh.

8.21.2.3 const_reference

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::const_reference = typename ContainerType::const_reference
```

Import the ContainerType::const_reference.

Definition at line 121 of file jeod_stl_container.hh.

8.21.2.4 const_reverse_iterator

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::const_reverse_iterator = typename
ContainerType::const_reverse_iterator
```

Import the ContainerType::const_reverse_iterator.

Definition at line 141 of file jeod_stl_container.hh.

8.21.2.5 difference_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::difference_type = typename ContainerType::
difference_type
```

Import the ContainerType::difference_type.

Definition at line 146 of file jeod_stl_container.hh.

8.21.2.6 iterator

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::iterator = typename ContainerType::
iterator
```

Import the ContainerType::iterator.

Definition at line 126 of file jeod_stl_container.hh.

8.21.2.7 reference

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::reference = typename ContainerType::
reference
```

Import the ContainerType::reference.

Definition at line 116 of file jeod_stl_container.hh.

8.21.2.8 reverse_iterator

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::reverse_iterator = typename ContainerType::reverse_iterator
```

Import the ContainerType::reverse_iterator.

Definition at line 136 of file jeod_stl_container.hh.

8.21.2.9 size_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::size_type = typename ContainerType::size_type
```

Import the ContainerType::size_type.

Definition at line 151 of file jeod_stl_container.hh.

8.21.2.10 this_container_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::this_container_type = JeodSTLContainer<ElemType, ContainerType>
```

This particular JeodSTLContainer type.

Definition at line 106 of file jeod_stl_container.hh.

8.21.2.11 value_type

```
template<typename ElemType, typename ContainerType>
using jeod::JeodSTLContainer< ElemType, ContainerType >::value_type = typename ContainerType::value_type
```

Import the ContainerType::value_type.

Definition at line 156 of file jeod_stl_container.hh.

8.21.3 Constructor & Destructor Documentation

8.21.3.1 ~JeodSTLContainer()

```
template<typename ElemType, typename ContainerType>
virtual jeod::JeodSTLContainer< ElemType, ContainerType >::~~JeodSTLContainer ( ) [virtual],
[default]
```

Destructor.

8.21.3.2 JeodSTLContainer() [1/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSTLContainer< ElemType, ContainerType >::~JeodSTLContainer ( ) [protected], [default]
```

Default constructor.

Note: Making this protected precludes someone from declaring an object to be of type JEODSTLContainer. Access is via some other class that inherits from this class.

8.21.3.3 JeodSTLContainer() [2/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSTLContainer< ElemType, ContainerType >::~JeodSTLContainer (
    const this_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 362 of file jeod_stl_container.hh.

8.21.3.4 JeodSTLContainer() [3/3]

```
template<typename ElemType, typename ContainerType>
jeod::JeodSTLContainer< ElemType, ContainerType >::~JeodSTLContainer (
    const ContainerType & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

<i>src</i>	Source container to be copied
------------	-------------------------------

Definition at line 371 of file jeod_stl_container.hh.

8.21.4 Member Function Documentation

8.21.4.1 begin() [1/2]

```
template<typename ElemType, typename ContainerType>
iterator jeod::JeodSTLContainer< ElemType, ContainerType >::begin ( ) [inline]
```

Returns an iterator that points to the first element.

Definition at line 236 of file jeod_stl_container.hh.

8.21.4.2 begin() [2/2]

```
template<typename ElemType, typename ContainerType>
const_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::begin ( ) const [inline]
```

Returns a const iterator that points to the first element.

Definition at line 244 of file jeod_stl_container.hh.

8.21.4.3 clear()

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSTLContainer< ElemType, ContainerType >::clear ( ) [inline]
```

Clear the contents.

Definition at line 328 of file jeod_stl_container.hh.

Referenced by jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::operator=().

8.21.4.4 empty()

```
template<typename ElemType, typename ContainerType>
bool jeod::JeodSTLContainer< ElemType, ContainerType >::empty ( ) const [inline]
```

Returns true if the contents are empty, false otherwise.

Definition at line 302 of file jeod_stl_container.hh.

8.21.4.5 `end()` [1/2]

```
template<typename ElemType, typename ContainerType>
iterator jeod::JeodSTLContainer< ElemType, ContainerType >::end ( ) [inline]
```

Returns an iterator that points past the last element.

Definition at line 252 of file `jeod_stl_container.hh`.

8.21.4.6 `end()` [2/2]

```
template<typename ElemType, typename ContainerType>
const_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::end ( ) const [inline]
```

Returns a const iterator that points past the last element.

Definition at line 260 of file `jeod_stl_container.hh`.

8.21.4.7 `get_allocator()`

```
template<typename ElemType, typename ContainerType>
allocator_type jeod::JeodSTLContainer< ElemType, ContainerType >::get_allocator ( ) const
[inline]
```

Returns the allocator object used to construct the contents.

Definition at line 226 of file `jeod_stl_container.hh`.

8.21.4.8 `insert()`

```
template<typename ElemType, typename ContainerType>
iterator jeod::JeodSTLContainer< ElemType, ContainerType >::insert (
    iterator position,
    const value_type & new_elem ) [inline]
```

Insert a new element initialized with *new_elem* before the iterator *position*.

Parameters

<i>position</i>	Insertion position
<i>new_elem</i>	Element value to be inserted

Returns

Iterator that points to the newly-inserted element

Definition at line 340 of file jeod_stl_container.hh.

8.21.4.9 max_size()

```
template<typename ElemType, typename ContainerType>
size_type jeod::JeodSTLContainer< ElemType, ContainerType >::max_size ( ) const [inline]
```

Returns the implementation's limit on the number of elements.

Definition at line 310 of file jeod_stl_container.hh.

8.21.4.10 operator const ContainerType &()

```
template<typename ElemType, typename ContainerType>
jeod::JeodSTLContainer< ElemType, ContainerType >::operator const ContainerType & ( ) const
[inline]
```

Returns the contents as a const rvalue.

Definition at line 186 of file jeod_stl_container.hh.

8.21.4.11 operator ContainerType &()

```
template<typename ElemType, typename ContainerType>
jeod::JeodSTLContainer< ElemType, ContainerType >::operator ContainerType & ( ) [inline]
```

Returns the contents as an lvalue.

Definition at line 178 of file jeod_stl_container.hh.

8.21.4.12 operator=() [1/2]

```
template<typename ElemType, typename ContainerType>
this_container_type& jeod::JeodSTLContainer< ElemType, ContainerType >::operator= (
    const this_container_type & src ) [inline]
```

Assignment operator.

Parameters

<code>src</code>	Source container to be copied
------------------	-------------------------------

Definition at line 197 of file `jeod_stl_container.hh`.

8.21.4.13 operator=() [2/2]

```
template<typename ElemType, typename ContainerType>
this_container_type& jeod::JeodSTLContainer< ElemType, ContainerType >::operator= (
    const ContainerType & src ) [inline]
```

Assignment operator.

Parameters

<code>src</code>	Source container to be copied
------------------	-------------------------------

Definition at line 211 of file `jeod_stl_container.hh`.

8.21.4.14 rbegin() [1/2]

```
template<typename ElemType, typename ContainerType>
reverse_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::rbegin ( ) [inline]
```

Returns a reverse iterator that points to the last element.

Definition at line 268 of file `jeod_stl_container.hh`.

8.21.4.15 rbegin() [2/2]

```
template<typename ElemType, typename ContainerType>
const_reverse_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::rbegin ( ) const
[inline]
```

Returns a const reverse iterator that points to the last element.

Definition at line 276 of file `jeod_stl_container.hh`.

8.21.4.16 `rend()` [1/2]

```
template<typename ElemType, typename ContainerType>
reverse_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::rend ( ) [inline]
```

Returns a reverse iterator that points before the first element.

Definition at line 284 of file jeod_stl_container.hh.

8.21.4.17 `rend()` [2/2]

```
template<typename ElemType, typename ContainerType>
const_reverse_iterator jeod::JeodSTLContainer< ElemType, ContainerType >::rend ( ) const
[inline]
```

Returns a const reverse iterator that points before the first element.

Definition at line 292 of file jeod_stl_container.hh.

8.21.4.18 `size()`

```
template<typename ElemType, typename ContainerType>
size_type jeod::JeodSTLContainer< ElemType, ContainerType >::size ( ) const [inline]
```

Returns the number of elements.

Definition at line 318 of file jeod_stl_container.hh.

8.21.4.19 `swap()` [1/2]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSTLContainer< ElemType, ContainerType >::swap (
    this_container_type & other ) [inline], [protected]
```

Swap contents.

Parameters

<i>other</i>	Other JEOD container with contents are to be swapped.
--------------	---

Definition at line 384 of file jeod_stl_container.hh.

8.21.4.20 swap() [2/2]

```
template<typename ElemType, typename ContainerType>
void jeod::JeodSTLContainer< ElemType, ContainerType >::swap (
    ContainerType & other ) [inline], [protected]
```

Swap contents.

Parameters

<i>other</i>	Other STL container with contents are to be swapped.
--------------	--

Definition at line 393 of file jeod_stl_container.hh.

8.21.5 Field Documentation

8.21.5.1 contents

```
template<typename ElemType, typename ContainerType>
ContainerType jeod::JeodSTLContainer< ElemType, ContainerType >::contents [protected]
```

The STL container.

trick_io(**)

Definition at line 403 of file jeod_stl_container.hh.

Referenced by jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::assign(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::back(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::begin(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::clear(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::count(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::empty(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::end(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::equal_range(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::erase(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::erase(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::find(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::front(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::get_allocator(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::insert(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::insert(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::insert(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::key_comp(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::lower_bound(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::max_size(), jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::operator const std::vector< ElemType > &(), jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::operator std::vector< ElemType > &(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::operator=(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::pop_back(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::push_back(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::rbegin(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::rend(), jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >::resize(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::size(), jeod::JeodSTLContainer< ElemType, std::list< ElemType > >::swap(), jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::upper_bound(), and jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >::value_comp().

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

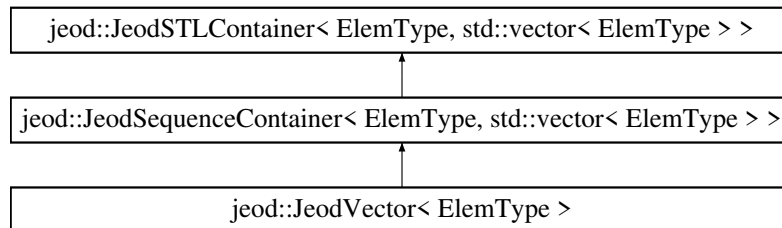
- [jeod_stl_container.hh](#)

8.22 jeod::JeodVector< ElemType > Class Template Reference

The JEOD replacement for `std::vector`.

```
#include <jeod_vector.hh>
```

Inheritance diagram for `jeod::JeodVector< ElemType >`:



Public Types

- using `this_container_type` = `JeodVector< ElemType >`
This particular `JeodVector` type.
- using `jeod_sequence_container_type` = `JeodSequenceContainer< ElemType, std::vector< ElemType > >`
The `JeodSequenceContainer` type.
- using `jeod_stl_container_type` = `JeodSTLContainer< ElemType, std::vector< ElemType > >`
The `JeodSTLContainer` type.
- using `stl_container_type` = `std::vector< ElemType >`
The `std::vector` itself.

Public Member Functions

- virtual `~JeodVector` ()=default
Destructor.
- `JeodVector` & `operator=` (const `this_container_type` &src)
Copy contents from the given source.
- `JeodVector` & `operator=` (const `stl_container_type` &src)
Copy contents from the given source.
- `jeod_stl_container_type::size_type` `capacity` () const
Returns the size of the allocated storage space for the vector.
- void `reserve` (typename `jeod_stl_container_type::size_type` n)
Requests that the capacity of the allocated storage space be made large enough to hold at least n elements.
- `stl_container_type::reference` `operator[]` (std::size_t n)
Get the nth element of the vector.
- `stl_container_type::const_reference` `operator[]` (std::size_t n) const
Get the nth element of the vector.
- `stl_container_type::reference` `at` (std::size_t n)
Get the nth element of the vector, throwing exception if out of range.
- `stl_container_type::const_reference` `at` (std::size_t n) const
Get the nth element of the vector, throwing exception if out of range.

Protected Member Functions

- [JeodVector](#) ()=default
Default constructor.
- [JeodVector](#) (const [this_container_type](#) &src)
Copy constructor.
- [JeodVector](#) (const [stl_container_type](#) &src)
Copy constructor from STL container.

Additional Inherited Members

8.22.1 Detailed Description

```
template<typename ElemType>
class jeod::JeodVector< ElemType >
```

The JEOD replacement for std::vector.

Definition at line 87 of file jeod_vector.hh.

8.22.2 Member Typedef Documentation

8.22.2.1 jeod_sequence_container_type

```
template<typename ElemType >
using jeod::JeodVector< ElemType >::jeod_sequence_container_type = JeodSequenceContainer<ElemType,
Type, std::vector<ElemType> >
```

The [JeodSequenceContainer](#) type.

Definition at line 100 of file jeod_vector.hh.

8.22.2.2 jeod_stl_container_type

```
template<typename ElemType >
using jeod::JeodVector< ElemType >::jeod_stl_container_type = JeodSTLContainer<ElemType,
std::vector<ElemType> >
```

The [JeodSTLContainer](#) type.

Definition at line 105 of file jeod_vector.hh.

8.22.2.3 stl_container_type

```
template<typename ElemType >
using jeod::JeodVector< ElemType >::stl_container_type = std::vector<ElemType>
```

The std::vector itself.

Definition at line 110 of file jeod_vector.hh.

8.22.2.4 this_container_type

```
template<typename ElemType >
using jeod::JeodVector< ElemType >::this_container_type = JeodVector<ElemType>
```

This particular JeodVector type.

Definition at line 95 of file jeod_vector.hh.

8.22.3 Constructor & Destructor Documentation

8.22.3.1 ~JeodVector()

```
template<typename ElemType >
virtual jeod::JeodVector< ElemType >::~~JeodVector ( ) [virtual], [default]
```

Destructor.

8.22.3.2 JeodVector() [1/3]

```
template<typename ElemType >
jeod::JeodVector< ElemType >::JeodVector ( ) [protected], [default]
```

Default constructor.

8.22.3.3 JeodVector() [2/3]

```
template<typename ElemType >
jeod::JeodVector< ElemType >::JeodVector (
    const this_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor.

Definition at line 208 of file jeod_vector.hh.

8.22.3.4 JeodVector() [3/3]

```
template<typename ElemType >
jeod::JeodVector< ElemType >::JeodVector (
    const stl_container_type & src ) [inline], [explicit], [protected]
```

Copy constructor from STL container.

Parameters

src	Source container to be copied
-----	-------------------------------

Definition at line 217 of file jeod_vector.hh.

8.22.4 Member Function Documentation**8.22.4.1 at()** [1/2]

```
template<typename ElemType >
stl_container_type::reference jeod::JeodVector< ElemType >::at (
    std::size_t n ) [inline]
```

Get the nth element of the vector, throwing exception if out of range.

Returns

Nth element of the vector.

Definition at line 185 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents.

8.22.4.2 at() [2/2]

```
template<typename ElemType >
stl_container_type::const_reference jeod::JeodVector< ElemType >::at (
    std::size_t n ) const [inline]
```

Get the nth element of the vector, throwing exception if out of range.

Returns

Nth element of the vector.

Definition at line 194 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents.

8.22.4.3 capacity()

```
template<typename ElemType >
jeod_stl_container_type::size_type jeod::JeodVector< ElemType >::capacity ( ) const [inline]
```

Returns the size of the allocated storage space for the vector.

Definition at line 147 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents.

8.22.4.4 operator=() [1/2]

```
template<typename ElemType >
JeodVector& jeod::JeodVector< ElemType >::operator= (
    const this_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 127 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::operator=().

8.22.4.5 operator=() [2/2]

```
template<typename ElemType >
JeodVector& jeod::JeodVector< ElemType >::operator= (
    const stl_container_type & src ) [inline]
```

Copy contents from the given source.

Definition at line 136 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::operator=().

8.22.4.6 operator[]() [1/2]

```
template<typename ElemType >
stl_container_type::reference jeod::JeodVector< ElemType >::operator[] (
    std::size_t n ) [inline]
```

Get the nth element of the vector.

Returns

Nth element of the vector.

Definition at line 167 of file jeod_vector.hh.

References jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents.

8.22.4.7 `operator[]()` [2/2]

```
template<typename ElemType >
stl_container_type::const_reference jeod::JeodVector< ElemType >::operator[] (
    std::size_t n ) const [inline]
```

Get the n th element of the vector.

Returns

Nth element of the vector.

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

References `jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents`.

8.22.4.8 `reserve()`

```
template<typename ElemType >
void jeod::JeodVector< ElemType >::reserve (
    typename jeod_stl_container_type::size_type n ) [inline]
```

Requests that the capacity of the allocated storage space be made large enough to hold at least n elements.

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

References `jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >::contents`.

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

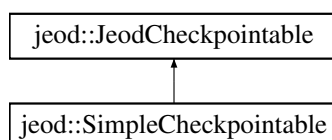
- [jeod_vector.hh](#)

8.23 `jeod::SimpleCheckpointable` Class Reference

The `SimpleCheckpointable` class provides a simple checkpoint/restart interface by which an object can complete the restart process.

```
#include <simple_checkpointable.hh>
```

Inheritance diagram for `jeod::SimpleCheckpointable`:



Public Member Functions

- [SimpleCheckpointable](#) ()=default
Construct a [SimpleCheckpointable](#) object.
- [~SimpleCheckpointable](#) () override=default
Destruct a [SimpleCheckpointable](#) object.
- const std::string [get_init_name](#) () override
Return the name of the initial restart action, in this case "restore".
- const std::string [get_item_name](#) () override
Return the name of the current restart action, in this case "".
- const std::string [get_item_value](#) () override
Return the value of the current restart action, in this case "".
- void [start_checkpoint](#) () override
In general, start the checkpoint process.
- void [advance_checkpoint](#) () override
In general, advance to the next checkpoint item; in this case, do nothing.
- bool [is_checkpoint_finished](#) () override
In general, indicate when checkpointing is complete.
- int [perform_restore_action](#) (const std::string &action_name, const std::string &action_value) override
In general, respond to the actions recorded in the checkpoint file.
- [SimpleCheckpointable](#) (const [SimpleCheckpointable](#) &)=delete
- [SimpleCheckpointable](#) & [operator=](#) (const [SimpleCheckpointable](#) &)=delete

Protected Member Functions

- virtual void [simple_restore](#) ()=0
Perform the sole restore action.

Friends

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

8.23.1 Detailed Description

The [SimpleCheckpointable](#) class provides a simple checkpoint/restart interface by which an object can complete the restart process.

Typical use of the class is to restore inherently uncheckpointable data such as file streams and function pointers.

The [SimpleCheckpointable](#) is an incomplete class. Derived classes must define a [simple_restore\(\)](#) method to make the derived class complete. This method will be called as a part of the container restart process. Those derived classes should not override the overrides provided by this class. Derived classes can override the `pre_` and `post_` checkpoint and restart methods.

Definition at line 85 of file `simple_checkpointable.hh`.

8.23.2 Constructor & Destructor Documentation

8.23.2.1 SimpleCheckpointable() [1/2]

```
jeod::SimpleCheckpointable::SimpleCheckpointable ( ) [default]
```

Construct a [SimpleCheckpointable](#) object.

8.23.2.2 ~SimpleCheckpointable()

```
jeod::SimpleCheckpointable::~~SimpleCheckpointable ( ) [override], [default]
```

Destruct a [SimpleCheckpointable](#) object.

8.23.2.3 SimpleCheckpointable() [2/2]

```
jeod::SimpleCheckpointable::SimpleCheckpointable (
    const SimpleCheckpointable & ) [delete]
```

8.23.3 Member Function Documentation**8.23.3.1 advance_checkpoint()**

```
void jeod::SimpleCheckpointable::advance_checkpoint ( ) [inline], [override], [virtual]
```

In general, advance to the next checkpoint item; in this case, do nothing.

This method is not called because the class immediately designates the checkpoint to be finished.

Implements [jeod::JeodCheckpointable](#).

Definition at line 138 of file simple_checkpointable.hh.

8.23.3.2 get_init_name()

```
const std::string jeod::SimpleCheckpointable::get_init_name ( ) [inline], [override], [virtual]
```

Return the name of the initial restart action, in this case "restore".

A derived class can of course override this.

Implements [jeod::JeodCheckpointable](#).

Definition at line 102 of file simple_checkpointable.hh.

8.23.3.3 get_item_name()

```
const std::string jeod::SimpleCheckpointable::get_item_name ( ) [inline], [override], [virtual]
```

Return the name of the current restart action, in this case "".

This method is not called because the class immediately designates the checkpoint to be finished.

Implements [jeod::JeodCheckpointable](#).

Definition at line 112 of file simple_checkpointable.hh.

8.23.3.4 get_item_value()

```
const std::string jeod::SimpleCheckpointable::get_item_value ( ) [inline], [override], [virtual]
```

Return the value of the current restart action, in this case "".

This method is not called because the class immediately designates the checkpoint to be finished.

Implements [jeod::JeodCheckpointable](#).

Definition at line 122 of file simple_checkpointable.hh.

8.23.3.5 is_checkpoint_finished()

```
bool jeod::SimpleCheckpointable::is_checkpoint_finished ( ) [inline], [override], [virtual]
```

In general, indicate when checkpointing is complete.

For this class, always return true.

Implements [jeod::JeodCheckpointable](#).

Definition at line 144 of file simple_checkpointable.hh.

8.23.3.6 operator=()

```
SimpleCheckpointable& jeod::SimpleCheckpointable::operator= (
    const SimpleCheckpointable & ) [delete]
```

8.23.3.7 perform_restore_action()

```
int jeod::SimpleCheckpointable::perform_restore_action (
    const std::string & action_name,
    const std::string & action_value ) [inline], [override], [virtual]
```

In general, respond to the actions recorded in the checkpoint file.

For this class, the only recorded action is "restore", and the response is to invoke the (undefined) simple_restore method.

Parameters

<i>action_name</i>	The name of the action; here just "restore".
<i>action_value</i>	The value of the action; here ignored.

Returns

Success (zero) / failure (non-zero).

Implements [jeod::JeodCheckpointable](#).

Definition at line 157 of file simple_checkpointable.hh.

8.23.3.8 simple_restore()

```
virtual void jeod::SimpleCheckpointable::simple_restore ( ) [protected], [pure virtual]
```

Perform the sole restore action.

8.23.3.9 start_checkpoint()

```
void jeod::SimpleCheckpointable::start_checkpoint ( ) [inline], [override], [virtual]
```

In general, start the checkpoint process.

For this class, do nothing.

Implements [jeod::JeodCheckpointable](#).

Definition at line 131 of file simple_checkpointable.hh.

8.23.4 Friends And Related Function Documentation**8.23.4.1 init_attrjeod__SimpleCheckpointable**

```
void init_attrjeod__SimpleCheckpointable ( ) [friend]
```

8.23.4.2 InputProcessor

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

Definition at line 87 of file simple_checkpointable.hh.

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

- [simple_checkpointable.hh](#)

Chapter 9

File Documentation

9.1 checkpointable.hh File Reference

Define the class JeodCheckpointable, the base class for checkpointing and restoring data that are opaque to the simulation engine.

```
#include <string>
#include <typeinfo>
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodCheckpointable](#)

A [JeodCheckpointable](#) is an object whose contents are opaque to Trick, and presumably other simulation engines, whose contents can nonetheless be checkpointed and restarted by using the methods defined herein.

Namespaces

- [jeod](#)

Namespace jeod.

9.1.1 Detailed Description

Define the class JeodCheckpointable, the base class for checkpointing and restoring data that are opaque to the simulation engine.

9.2 container.hh File Reference

Define the class JeodContainer, which adds checkpointability to an STL sequence container replacement.

```
#include "checkpointable.hh"
#include "utils/memory/include/memory_manager.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <cstring>
#include <string>
#include <typeinfo>
```

Data Structures

- class [jeod::JeodContainer< ContainerType, ElemType >](#)
A [JeodContainer](#) is a JEOD STL sequence container replacement whose contents are checkpointable and restorable.

Namespaces

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

9.2.1 Detailed Description

Define the class JeodContainer, which adds checkpointability to an STL sequence container replacement.

9.3 jeod_associative_container.hh File Reference

Define checkpointable replacements for STL associative containers.

```
#include "jeod_stl_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <utility>
```

Data Structures

- class [jeod::JeodAssociativeContainer< ElemType, ContainerType >](#)
This is the base class for the JEOD replacements of the STL associative containers.

Namespaces

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

9.3.1 Detailed Description

Define checkpointable replacements for STL associative containers.

This file defines class template JeodAssociativeContainer, the basis for the concept. The ultimate goal is to provide the full functionality of the ISO/IEC 14882:2003 STL associative containers as transparently as possible in the form of checkpointable class templates.

9.4 jeod_container_compare.hh File Reference

Define comparison operators for JEOD STL container.

```
#include "jeod_stl_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Functions

- `template<typename ElemType , typename ContainerType >`
`bool operator< (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator< (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator< (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is less than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator== (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator> (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is greater than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator>= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is greater than or equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator!= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`
Test if x is not equal to y.
- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const ContainerType &y)`

Test if x is less than or equal to y.

- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const ContainerType &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`

Test if x is less than or equal to y.

- `template<typename ElemType , typename ContainerType >`
`bool operator<= (const jeod::JeodSTLContainer< ElemType, ContainerType > &x, const jeod::JeodSTLContainer< ElemType, ContainerType > &y)`

Test if x is less than or equal to y.

9.4.1 Detailed Description

Define comparison operators for JEOD STL container.

The comparisons are the same as those for the underlying STL containers and are implemented using the underlying STL container comparison operators. There are three template functions to define for each comparison operator:

- JEOD container to STL container
- STL container to JEOD container
- JEOD container to JEOD container. With 6 comparison operators this means 18 function templates need to be defined.

9.5 jeod_list.hh File Reference

Define the class template JeodList.

```
#include "jeod_sequence_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <list>
```

Data Structures

- class `jeod::JeodList< ElemType >`
The JEOD replacement for std::list.

Namespaces

- `jeod`
Namespace jeod.

9.5.1 Detailed Description

Define the class template JeodList.

9.6 jeod_sequence_container.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_stl_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodSequenceContainer< ElemType, ContainerType >](#)
This is the base class for the JEOD replacements of the STL sequence containers.

Namespaces

- [jeod](#)
Namespace jeod.

9.6.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

This file defines class template JeodSequenceContainer, the basis for the concept. The ultimate goal is to provide the full functionality of the ISO/IEC 14882:2003 STL sequence containers as transparently as possible in the form of checkpointable class templates.

9.7 jeod_set.hh File Reference

Define the class template JeodSet.

```
#include "jeod_associative_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <set>
```

Data Structures

- class [jeod::JeodSet< ElemType >](#)
The JEOD replacement for std::set.

Namespaces

- [jeod](#)
Namespace jeod.

9.7.1 Detailed Description

Define the class template JeodSet.

9.8 jeod_stl_container.hh File Reference

Define checkpointable replacements for STL containers.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "jeod_container_compare.hh"
```

Data Structures

- class [jeod::JeodSTLContainer< ElemType, ContainerType >](#)
This is the base class for the JEOD replacements of the STL containers.

Namespaces

- [jeod](#)
Namespace jeod.

9.8.1 Detailed Description

Define checkpointable replacements for STL containers.

This file defines class template JeodSTLContainer, the starting point of this concept. The ultimate goal is to provide the full functionality of the ISO/IEC 14882:2003 STL containers as transparently as possible in the form of checkpointable class templates.

9.9 jeod_vector.hh File Reference

Define class template JeodVector.

```
#include "jeod_sequence_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <cstdlib>
#include <vector>
```

Data Structures

- class [jeod::JeodVector< ElemType >](#)
The JEOD replacement for std::vector.

Namespaces

- [jeod](#)

Namespace jeod.

9.9.1 Detailed Description

Define class template JeodVector.

9.10 object_container.hh File Reference

Define class template JeodObjectContainer.

```
#include "container.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/sim_interface/include/config.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/sim_interface/include/simulation_interface.hh"
#include <cstdint>
#include <string>
```

Data Structures

- class [jeod::JeodObjectContainer< ContainerType, ElemType >](#)
A [JeodObjectContainer](#) is a [JeodContainer](#) that contains objects of type ElemType.

Namespaces

- [jeod](#)

Namespace jeod.

Macros

- `#define JEOD_OBJECT_CONTAINER(container_type, elem_type) JeodObjectContainer<Jeod##container↵_type<elem_type>, elem_type>`

9.10.1 Detailed Description

Define class template JeodObjectContainer.

9.10.2 Macro Definition Documentation

9.10.2.1 JEOD_OBJECT_CONTAINER

```
#define JEOD_OBJECT_CONTAINER(  
    container_type,  
    elem_type ) JeodObjectContainer<Jeod##container_type<elem_type>, elem_type>
```

Definition at line 280 of file object_container.hh.

9.11 object_list.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_list.hh"  
#include "object_container.hh"  
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodObjectList< ElemType >](#)
Defines a registry for defining a checkpointable list of objects.

Namespaces

- [jeod](#)
Namespace jeod.

9.11.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.12 object_set.hh File Reference

Define checkpointable replacements for STL associative containers.

```
#include "jeod_set.hh"  
#include "object_container.hh"  
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodObjectSet< ElemType >](#)
Defines a registry for defining a checkpointable set of objects.

Namespaces

- [jeod](#)

Namespace jeod.

9.12.1 Detailed Description

Define checkpointable replacements for STL associative containers.

9.13 object_vector.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_vector.hh"
#include "object_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodObjectVector< ElemType >](#)

Defines a registry for defining a checkpointable vector of objects.

Namespaces

- [jeod](#)

Namespace jeod.

9.13.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.14 pointer_container.hh File Reference

Define class template JeodPointerContainer.

```
#include "container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/sim_interface/include/simulation_interface.hh"
#include <string>
```

Data Structures

- class [jeod::JeodPointerContainer](#)< ContainerType, ElemType >

A *JeodPointerContainer* is a *JeodContainer* that contains pointers to objects of type *ElemType*.

Namespaces

- [jeod](#)

Namespace *jeod*.

Macros

- `#define JEOD_POINTER_CONTAINER(container_type, elem_type) JeodPointerContainer<Jeod##container_type<elem_type *>, elem_type>`

9.14.1 Detailed Description

Define class template `JeodPointerContainer`.

9.14.2 Macro Definition Documentation

9.14.2.1 JEOD_POINTER_CONTAINER

```
#define JEOD_POINTER_CONTAINER(  
    container_type,  
    elem_type ) JeodPointerContainer<Jeod##container_type<elem_type *>, elem_type>
```

Definition at line 194 of file `pointer_container.hh`.

9.15 pointer_list.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_list.hh"  
#include "pointer_container.hh"  
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPointerList](#)< ElemType >

Defines a registry for defining a checkpointable list of pointers.

Namespaces

- [jeod](#)

Namespace jeod.

9.15.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.16 pointer_set.hh File Reference

Define checkpointable replacements for STL associative containers.

```
#include "jeod_set.hh"
#include "pointer_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPointerSet< ElemType >](#)

Defines a registry for defining a checkpointable set of pointers.

Namespaces

- [jeod](#)

Namespace jeod.

9.16.1 Detailed Description

Define checkpointable replacements for STL associative containers.

9.17 pointer_vector.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_vector.hh"
#include "pointer_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPointerVector< ElemType >](#)

Defines a registry for defining a checkpointable vector of pointers.

Namespaces

- [jeod](#)

Namespace jeod.

9.17.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.18 primitive_container.hh File Reference

Define class template JeodPrimitiveContainer.

```
#include "container.hh"
#include "primitive_serializer.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <string>
```

Data Structures

- class [jeod::JeodPrimitiveContainer< ContainerType, ElemType >](#)
*A [JeodPrimitiveContainer](#) is a [JeodContainer](#) that contains primitive data of type *ElemType*.*

Namespaces

- [jeod](#)

Namespace jeod.

Macros

- `#define JEOD_PRIMITIVE_CONTAINER(container_type, elem_type) JeodPrimitiveContainer<Jeod##container_type<elem_type>, elem_type>`

9.18.1 Detailed Description

Define class template JeodPrimitiveContainer.

9.18.2 Macro Definition Documentation

9.18.2.1 JEOD_PRIMITIVE_CONTAINER

```
#define JEOD_PRIMITIVE_CONTAINER(  
    container_type,  
    elem_type ) JeodPrimitiveContainer<Jeod##container_type<elem_type>, elem_type>
```

Definition at line 169 of file primitive_container.hh.

9.19 primitive_list.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_list.hh"  
#include "primitive_container.hh"  
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPrimitiveList< ElemType >](#)
Defines a registry for defining a checkpointable list of primitives.

Namespaces

- [jeod](#)
Namespace jeod.

9.19.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.20 primitive_serializer.cc File Reference

Define class JeodPrimitiveSerializerBase static methods.

```
#include <cmath>  
#include <cstdlib>  
#include <limits>  
#include <sstream>  
#include <string>  
#include "../include/primitive_serializer.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

Macros

- `#define __USE_ISOC99`

9.20.1 Detailed Description

Define class JeodPrimitiveSerializerBase static methods.

9.21 primitive_serializer.hh File Reference

Define class template JeodPrimitiveSerializer.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include <cmath>
#include <limits>
#include <sstream>
#include <string>
```

Data Structures

- class `jeod::JeodPrimitiveSerializerBase`
Base class for serializing / deserializing primitive data.
- class `jeod::JeodPrimitiveSerializer< Type >`
Serializer / deserializer for primitive data.

Namespaces

- `jeod`
Namespace jeod.

9.21.1 Detailed Description

Define class template JeodPrimitiveSerializer.

9.22 primitive_set.hh File Reference

Define checkpointable replacements for STL associative containers.

```
#include "jeod_set.hh"
#include "primitive_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPrimitiveSet< ElemType >](#)
Defines a registry for defining a checkpointable set of primitives.

Namespaces

- [jeod](#)
Namespace jeod.

9.22.1 Detailed Description

Define checkpointable replacements for STL associative containers.

9.23 primitive_vector.hh File Reference

Define checkpointable replacements for STL sequence containers.

```
#include "jeod_vector.hh"
#include "primitive_container.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::JeodPrimitiveVector< ElemType >](#)
Defines a registry for defining a checkpointable vector of primitives.

Namespaces

- [jeod](#)
Namespace jeod.

9.23.1 Detailed Description

Define checkpointable replacements for STL sequence containers.

9.24 simple_checkpointable.hh File Reference

Define the class SimpleCheckpointable.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "checkpointable.hh"
```

Data Structures

- class [jeod::SimpleCheckpointable](#)

The [SimpleCheckpointable](#) class provides a simple checkpoint/restart interface by which an object can complete the restart process.

Namespaces

- [jeod](#)

Namespace [jeod](#).

9.24.1 Detailed Description

Define the class SimpleCheckpointable.

Index

- `__USE_ISOC99`
 - Container, [15](#)
- `~JeodAssociativeContainer`
 - `jeod::JeodAssociativeContainer`, [30](#)
- `~JeodCheckpointable`
 - `jeod::JeodCheckpointable`, [38](#)
- `~JeodContainer`
 - `jeod::JeodContainer`, [47](#)
- `~JeodList`
 - `jeod::JeodList`, [56](#)
- `~JeodObjectContainer`
 - `jeod::JeodObjectContainer`, [65](#)
- `~JeodPointerContainer`
 - `jeod::JeodPointerContainer`, [75](#)
- `~JeodPrimitiveContainer`
 - `jeod::JeodPrimitiveContainer`, [82](#)
- `~JeodPrimitiveSerializer`
 - `jeod::JeodPrimitiveSerializer`, [86](#)
- `~JeodPrimitiveSerializerBase`
 - `jeod::JeodPrimitiveSerializerBase`, [91](#)
- `~JeodSTLContainer`
 - `jeod::JeodSTLContainer`, [113](#)
- `~JeodSequenceContainer`
 - `jeod::JeodSequenceContainer`, [99](#)
- `~JeodSet`
 - `jeod::JeodSet`, [107](#)
- `~JeodVector`
 - `jeod::JeodVector`, [123](#)
- `~SimpleCheckpointable`
 - `jeod::SimpleCheckpointable`, [128](#)
- `advance_checkpoint`
 - `jeod::JeodCheckpointable`, [38](#)
 - `jeod::JeodContainer`, [47](#)
 - `jeod::JeodObjectContainer`, [65](#)
 - `jeod::SimpleCheckpointable`, [128](#)
- `allocator_type`
 - `jeod::JeodSTLContainer`, [111](#)
- `assign`
 - `jeod::JeodSequenceContainer`, [100](#)
- `at`
 - `jeod::JeodVector`, [124](#)
- `back`
 - `jeod::JeodSequenceContainer`, [100](#), [101](#)
- `base_container_type`
 - `jeod::JeodAssociativeContainer`, [29](#)
 - `jeod::JeodSequenceContainer`, [98](#)
- `base_type_descriptor`
 - `jeod::JeodPointerContainer`, [77](#)
- `begin`
 - `jeod::JeodSTLContainer`, [115](#)
- `capacity`
 - `jeod::JeodVector`, [124](#)
- `checkpoint_iter`
 - `jeod::JeodContainer`, [53](#)
- `checkpointable.hh`, [131](#)
- `clear`
 - `jeod::JeodSTLContainer`, [115](#)
- `const_iterator`
 - `jeod::JeodSTLContainer`, [111](#)
- `const_reference`
 - `jeod::JeodSTLContainer`, [111](#)
- `const_reverse_iterator`
 - `jeod::JeodSTLContainer`, [111](#)
- `Container`, [13](#)
 - `__USE_ISOC99`, [15](#)
 - `operator!=`, [15](#), [16](#)
 - `operator<`, [17](#)
 - `operator<=`, [18](#), [19](#)
 - `operator>`, [20](#), [21](#)
 - `operator>=`, [22](#), [23](#)
 - `operator==`, [19](#), [20](#)
- `container.hh`, [131](#)
- `contents`
 - `jeod::JeodSTLContainer`, [120](#)
- `copy`
 - `jeod::JeodObjectContainer`, [69](#)
- `count`
 - `jeod::JeodAssociativeContainer`, [31](#)
- `deserialize_double`
 - `jeod::JeodPrimitiveSerializerBase`, [91](#)
- `deserialize_float`
 - `jeod::JeodPrimitiveSerializerBase`, [91](#)
- `deserialize_long_double`
 - `jeod::JeodPrimitiveSerializerBase`, [92](#)
- `deserialize_string`
 - `jeod::JeodPrimitiveSerializerBase`, [92](#)
- `difference_type`
 - `jeod::JeodSTLContainer`, [112](#)
- `elem_type_descriptor`
 - `jeod::JeodContainer`, [53](#)
- `empty`
 - `jeod::JeodSTLContainer`, [115](#)
- `end`
 - `jeod::JeodSTLContainer`, [115](#), [116](#)
- `equal_range`

- jeod::JeodAssociativeContainer, 32
- erase
 - jeod::JeodAssociativeContainer, 32, 33
 - jeod::JeodSequenceContainer, 101
- find
 - jeod::JeodAssociativeContainer, 33
- from_string
 - jeod::JeodPrimitiveSerializer, 87, 88
- front
 - jeod::JeodSequenceContainer, 102
- get_allocator
 - jeod::JeodSTLContainer, 116
- get_final_name
 - jeod::JeodCheckpointable, 38
 - jeod::JeodContainer, 47
- get_final_value
 - jeod::JeodCheckpointable, 39
 - jeod::JeodObjectContainer, 65
- get_init_name
 - jeod::JeodCheckpointable, 39
 - jeod::JeodContainer, 48
 - jeod::SimpleCheckpointable, 128
- get_init_value
 - jeod::JeodCheckpointable, 39
- get_item_name
 - jeod::JeodCheckpointable, 39
 - jeod::JeodContainer, 48
 - jeod::SimpleCheckpointable, 128
- get_item_value
 - jeod::JeodCheckpointable, 40
 - jeod::JeodObjectContainer, 66
 - jeod::JeodPointerContainer, 75
 - jeod::JeodPrimitiveContainer, 82
 - jeod::SimpleCheckpointable, 129
- index
 - jeod::JeodObjectContainer, 69
- init_attrjeod__JeodCheckpointable
 - jeod::JeodCheckpointable, 43
- init_attrjeod__JeodContainer
 - jeod::JeodContainer, 52
- init_attrjeod__JeodObjectContainer
 - jeod::JeodObjectContainer, 69
- init_attrjeod__SimpleCheckpointable
 - jeod::SimpleCheckpointable, 130
- initialize_checkpointable
 - jeod::JeodCheckpointable, 40
 - jeod::JeodContainer, 48
 - jeod::JeodPointerContainer, 75
- InputProcessor
 - jeod::JeodCheckpointable, 43
 - jeod::JeodContainer, 52
 - jeod::JeodObjectContainer, 69
 - jeod::SimpleCheckpointable, 130
- insert
 - jeod::JeodAssociativeContainer, 34
 - jeod::JeodSTLContainer, 116
- jeod::JeodSequenceContainer, 102, 103
- is_checkpoint_finished
 - jeod::JeodCheckpointable, 40
 - jeod::JeodContainer, 49
 - jeod::SimpleCheckpointable, 129
- iterator
 - jeod::JeodSTLContainer, 112
- JEOD_OBJECT_CONTAINER
 - object_container.hh, 137
- JEOD_POINTER_CONTAINER
 - pointer_container.hh, 140
- JEOD_PRIMITIVE_CONTAINER
 - primitive_container.hh, 142
- jeod, 25
- jeod::JeodAssociativeContainer
 - ~JeodAssociativeContainer, 30
 - base_container_type, 29
 - count, 31
 - equal_range, 32
 - erase, 32, 33
 - find, 33
 - insert, 34
 - JeodAssociativeContainer, 30, 31
 - key_comp, 35
 - key_compare, 29
 - key_type, 29
 - lower_bound, 35
 - this_container_type, 30
 - upper_bound, 35, 36
 - value_comp, 36
 - value_compare, 30
- jeod::JeodAssociativeContainer< ElemType, Container←
Type >, 27
- jeod::JeodCheckpointable, 36
 - ~JeodCheckpointable, 38
 - advance_checkpoint, 38
 - get_final_name, 38
 - get_final_value, 39
 - get_init_name, 39
 - get_init_value, 39
 - get_item_name, 39
 - get_item_value, 40
 - init_attrjeod__JeodCheckpointable, 43
 - initialize_checkpointable, 40
 - InputProcessor, 43
 - is_checkpoint_finished, 40
 - JeodCheckpointable, 38
 - operator=, 41
 - perform_restore_action, 41
 - post_checkpoint, 41
 - post_restart, 42
 - pre_checkpoint, 42
 - pre_restart, 42
 - start_checkpoint, 42
 - undo_initialize_checkpointable, 43
- jeod::JeodContainer
 - ~JeodContainer, 47
 - advance_checkpoint, 47

- checkpoint_iter, 53
- elem_type_descriptor, 53
- get_final_name, 47
- get_init_name, 48
- get_item_name, 48
- init_attrjeod__JeodContainer, 52
- initialize_checkpointable, 48
- InputProcessor, 52
- is_checkpoint_finished, 49
- JeodContainer, 46
- operator=, 49, 50
- perform_cleanup_action, 50
- perform_insert_action, 51
- perform_restore_action, 51
- start_checkpoint, 51
- stl_container_type, 45
- swap_contents, 52
- this_container_type, 45
- jeod::JeodContainer< ContainerType, ElemType >, 44
- jeod::JeodList
 - ~JeodList, 56
 - jeod_sequence_container_type, 55
 - jeod_stl_container_type, 56
 - JeodList, 57
 - merge, 57, 58
 - operator=, 58
 - pop_front, 59
 - push_front, 59
 - remove, 59
 - remove_if, 60
 - reverse, 60
 - sort, 60
 - splice, 61
 - stl_container_type, 56
 - this_container_type, 56
 - unique, 62
- jeod::JeodList< ElemType >, 54
- jeod::JeodObjectContainer
 - ~JeodObjectContainer, 65
 - advance_checkpoint, 65
 - copy, 69
 - get_final_value, 65
 - get_item_value, 66
 - index, 69
 - init_attrjeod__JeodObjectContainer, 69
 - InputProcessor, 69
 - JeodObjectContainer, 64, 65
 - operator=, 66, 67
 - perform_cleanup_action, 67
 - perform_insert_action, 67
 - post_checkpoint, 68
 - post_restart, 68
 - pre_checkpoint, 68
 - start_checkpoint, 68
- jeod::JeodObjectContainer< ContainerType, ElemType >, 63
- jeod::JeodObjectList
 - type, 70
- jeod::JeodObjectList< ElemType >, 70
- jeod::JeodObjectSet
 - type, 71
- jeod::JeodObjectSet< ElemType >, 71
- jeod::JeodObjectVector
 - type, 72
- jeod::JeodObjectVector< ElemType >, 72
- jeod::JeodPointerContainer
 - ~JeodPointerContainer, 75
 - base_type_descriptor, 77
 - get_item_value, 75
 - initialize_checkpointable, 75
 - JeodPointerContainer, 74
 - operator=, 75, 76
 - perform_insert_action, 76
- jeod::JeodPointerContainer< ContainerType, ElemType >, 73
- jeod::JeodPointerList
 - type, 78
- jeod::JeodPointerList< ElemType >, 77
- jeod::JeodPointerSet
 - type, 78
- jeod::JeodPointerSet< ElemType >, 78
- jeod::JeodPointerVector
 - type, 79
- jeod::JeodPointerVector< ElemType >, 79
- jeod::JeodPrimitiveContainer
 - ~JeodPrimitiveContainer, 82
 - get_item_value, 82
 - JeodPrimitiveContainer, 81
 - operator=, 82, 83
 - perform_insert_action, 83
 - serializer, 84
- jeod::JeodPrimitiveContainer< ContainerType, ElemType >, 80
- jeod::JeodPrimitiveList
 - type, 84
- jeod::JeodPrimitiveList< ElemType >, 84
- jeod::JeodPrimitiveSerializer
 - ~JeodPrimitiveSerializer, 86
 - from_string, 87, 88
 - JeodPrimitiveSerializer, 86
 - operator=, 88
 - to_string, 88, 89
- jeod::JeodPrimitiveSerializer< Type >, 85
- jeod::JeodPrimitiveSerializerBase, 90
 - ~JeodPrimitiveSerializerBase, 91
 - deserialize_double, 91
 - deserialize_float, 91
 - deserialize_long_double, 92
 - deserialize_string, 92
 - JeodPrimitiveSerializerBase, 90
 - serialize_double, 92
 - serialize_float, 93
 - serialize_long_double, 93
 - serialize_string, 94
- jeod::JeodPrimitiveSet
 - type, 95

jeod::JeodPrimitiveSet< ElemType >, 94
 jeod::JeodPrimitiveVector
 type, 96
 jeod::JeodPrimitiveVector< ElemType >, 95
 jeod::JeodSTLContainer
 ~JeodSTLContainer, 113
 allocator_type, 111
 begin, 115
 clear, 115
 const_iterator, 111
 const_reference, 111
 const_reverse_iterator, 111
 contents, 120
 difference_type, 112
 empty, 115
 end, 115, 116
 get_allocator, 116
 insert, 116
 iterator, 112
 JeodSTLContainer, 114
 max_size, 117
 operator const ContainerType &, 117
 operator ContainerType &, 117
 operator=, 117, 118
 rbegin, 118
 reference, 112
 rend, 118, 119
 reverse_iterator, 112
 size, 119
 size_type, 113
 swap, 119
 this_container_type, 113
 value_type, 113
 jeod::JeodSTLContainer< ElemType, ContainerType >, 108
 jeod::JeodSequenceContainer
 ~JeodSequenceContainer, 99
 assign, 100
 back, 100, 101
 base_container_type, 98
 erase, 101
 front, 102
 insert, 102, 103
 JeodSequenceContainer, 99
 pop_back, 103
 push_back, 104
 resize, 104
 this_container_type, 98
 jeod::JeodSequenceContainer< ElemType, ContainerType >, 96
 jeod::JeodSet
 ~JeodSet, 107
 jeod_associative_container_type, 106
 jeod_stl_container_type, 106
 JeodSet, 107
 operator=, 108
 stl_container_type, 106
 this_container_type, 106
 jeod::JeodSet< ElemType >, 104
 jeod::JeodVector
 ~JeodVector, 123
 at, 124
 capacity, 124
 jeod_sequence_container_type, 122
 jeod_stl_container_type, 122
 JeodVector, 123
 operator=, 125
 operator[], 125
 reserve, 126
 stl_container_type, 122
 this_container_type, 123
 jeod::JeodVector< ElemType >, 121
 jeod::SimpleCheckpointable, 126
 ~SimpleCheckpointable, 128
 advance_checkpoint, 128
 get_init_name, 128
 get_item_name, 128
 get_item_value, 129
 init_attrjeod_SimpleCheckpointable, 130
 InputProcessor, 130
 is_checkpoint_finished, 129
 operator=, 129
 perform_restore_action, 129
 simple_restore, 130
 SimpleCheckpointable, 127, 128
 start_checkpoint, 130
 jeod_associative_container.hh, 132
 jeod_associative_container_type
 jeod::JeodSet, 106
 jeod_container_compare.hh, 132
 jeod_list.hh, 134
 jeod_sequence_container.hh, 135
 jeod_sequence_container_type
 jeod::JeodList, 55
 jeod::JeodVector, 122
 jeod_set.hh, 135
 jeod_stl_container.hh, 136
 jeod_stl_container_type
 jeod::JeodList, 56
 jeod::JeodSet, 106
 jeod::JeodVector, 122
 jeod_vector.hh, 136
 JeodAssociativeContainer
 jeod::JeodAssociativeContainer, 30, 31
 JeodCheckpointable
 jeod::JeodCheckpointable, 38
 JeodContainer
 jeod::JeodContainer, 46
 JeodList
 jeod::JeodList, 57
 JeodObjectContainer
 jeod::JeodObjectContainer, 64, 65
 JeodPointerContainer
 jeod::JeodPointerContainer, 74
 JeodPrimitiveContainer
 jeod::JeodPrimitiveContainer, 81

- JeodPrimitiveSerializer
 - jeod::JeodPrimitiveSerializer, 86
- JeodPrimitiveSerializerBase
 - jeod::JeodPrimitiveSerializerBase, 90
- JeodSTLContainer
 - jeod::JeodSTLContainer, 114
- JeodSequenceContainer
 - jeod::JeodSequenceContainer, 99
- JeodSet
 - jeod::JeodSet, 107
- JeodVector
 - jeod::JeodVector, 123
- key_comp
 - jeod::JeodAssociativeContainer, 35
- key_compare
 - jeod::JeodAssociativeContainer, 29
- key_type
 - jeod::JeodAssociativeContainer, 29
- lower_bound
 - jeod::JeodAssociativeContainer, 35
- max_size
 - jeod::JeodSTLContainer, 117
- merge
 - jeod::JeodList, 57, 58
- Models, 11
- object_container.hh, 137
 - JEOD_OBJECT_CONTAINER, 137
- object_list.hh, 138
- object_set.hh, 138
- object_vector.hh, 139
- operator const ContainerType &
 - jeod::JeodSTLContainer, 117
- operator ContainerType &
 - jeod::JeodSTLContainer, 117
- operator!=
 - Container, 15, 16
- operator<
 - Container, 17
- operator<=
 - Container, 18, 19
- operator>
 - Container, 20, 21
- operator>=
 - Container, 22, 23
- operator=
 - jeod::JeodCheckpointable, 41
 - jeod::JeodContainer, 49, 50
 - jeod::JeodList, 58
 - jeod::JeodObjectContainer, 66, 67
 - jeod::JeodPointerContainer, 75, 76
 - jeod::JeodPrimitiveContainer, 82, 83
 - jeod::JeodPrimitiveSerializer, 88
 - jeod::JeodSTLContainer, 117, 118
 - jeod::JeodSet, 108
 - jeod::JeodVector, 125
 - jeod::SimpleCheckpointable, 129
- operator==
 - Container, 19, 20
- operator[]
 - jeod::JeodVector, 125
- perform_cleanup_action
 - jeod::JeodContainer, 50
 - jeod::JeodObjectContainer, 67
- perform_insert_action
 - jeod::JeodContainer, 51
 - jeod::JeodObjectContainer, 67
 - jeod::JeodPointerContainer, 76
 - jeod::JeodPrimitiveContainer, 83
- perform_restore_action
 - jeod::JeodCheckpointable, 41
 - jeod::JeodContainer, 51
 - jeod::SimpleCheckpointable, 129
- pointer_container.hh, 139
 - JEOD_POINTER_CONTAINER, 140
- pointer_list.hh, 140
- pointer_set.hh, 141
- pointer_vector.hh, 141
- pop_back
 - jeod::JeodSequenceContainer, 103
- pop_front
 - jeod::JeodList, 59
- post_checkpoint
 - jeod::JeodCheckpointable, 41
 - jeod::JeodObjectContainer, 68
- post_restart
 - jeod::JeodCheckpointable, 42
 - jeod::JeodObjectContainer, 68
- pre_checkpoint
 - jeod::JeodCheckpointable, 42
 - jeod::JeodObjectContainer, 68
- pre_restart
 - jeod::JeodCheckpointable, 42
- primitive_container.hh, 142
 - JEOD_PRIMITIVE_CONTAINER, 142
- primitive_list.hh, 143
- primitive_serializer.cc, 143
- primitive_serializer.hh, 144
- primitive_set.hh, 144
- primitive_vector.hh, 145
- push_back
 - jeod::JeodSequenceContainer, 104
- push_front
 - jeod::JeodList, 59
- rbegin
 - jeod::JeodSTLContainer, 118
- reference
 - jeod::JeodSTLContainer, 112
- remove
 - jeod::JeodList, 59
- remove_if
 - jeod::JeodList, 60
- rend

- jeod::JeodSTLContainer, 118, 119
- reserve
 - jeod::JeodVector, 126
- resize
 - jeod::JeodSequenceContainer, 104
- reverse
 - jeod::JeodList, 60
- reverse_iterator
 - jeod::JeodSTLContainer, 112
- serialize_double
 - jeod::JeodPrimitiveSerializerBase, 92
- serialize_float
 - jeod::JeodPrimitiveSerializerBase, 93
- serialize_long_double
 - jeod::JeodPrimitiveSerializerBase, 93
- serialize_string
 - jeod::JeodPrimitiveSerializerBase, 94
- serializer
 - jeod::JeodPrimitiveContainer, 84
- simple_checkpointable.hh, 145
- simple_restore
 - jeod::SimpleCheckpointable, 130
- SimpleCheckpointable
 - jeod::SimpleCheckpointable, 127, 128
- size
 - jeod::JeodSTLContainer, 119
- size_type
 - jeod::JeodSTLContainer, 113
- sort
 - jeod::JeodList, 60
- splice
 - jeod::JeodList, 61
- start_checkpoint
 - jeod::JeodCheckpointable, 42
 - jeod::JeodContainer, 51
 - jeod::JeodObjectContainer, 68
 - jeod::SimpleCheckpointable, 130
- stl_container_type
 - jeod::JeodContainer, 45
 - jeod::JeodList, 56
 - jeod::JeodSet, 106
 - jeod::JeodVector, 122
- swap
 - jeod::JeodSTLContainer, 119
- swap_contents
 - jeod::JeodContainer, 52
- this_container_type
 - jeod::JeodAssociativeContainer, 30
 - jeod::JeodContainer, 45
 - jeod::JeodList, 56
 - jeod::JeodSTLContainer, 113
 - jeod::JeodSequenceContainer, 98
 - jeod::JeodSet, 106
 - jeod::JeodVector, 123
- to_string
 - jeod::JeodPrimitiveSerializer, 88, 89
- type
 - jeod::JeodObjectList, 70
 - jeod::JeodObjectSet, 71
 - jeod::JeodObjectVector, 72
 - jeod::JeodPointerList, 78
 - jeod::JeodPointerSet, 78
 - jeod::JeodPointerVector, 79
 - jeod::JeodPrimitiveList, 84
 - jeod::JeodPrimitiveSet, 95
 - jeod::JeodPrimitiveVector, 96
- undo_initialize_checkpointable
 - jeod::JeodCheckpointable, 43
- unique
 - jeod::JeodList, 62
- upper_bound
 - jeod::JeodAssociativeContainer, 35, 36
- Utils, 12
- value_comp
 - jeod::JeodAssociativeContainer, 36
- value_compare
 - jeod::JeodAssociativeContainer, 30
- value_type
 - jeod::JeodSTLContainer, 113