

ContainerModel

5.0

Generated by Doxygen 1.8.5

Wed Jun 1 2022 12:10:25

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!=	15
6.3.3.2	operator!=	15
6.3.3.3	operator!=	16
6.3.3.4	operator<	16
6.3.3.5	operator<	16
6.3.3.6	operator<	17
6.3.3.7	operator<=	18
6.3.3.8	operator<=	18

6.3.3.9	operator<=	18
6.3.3.10	operator==	19
6.3.3.11	operator==	20
6.3.3.12	operator==	20
6.3.3.13	operator>	20
6.3.3.14	operator>	21
6.3.3.15	operator>	22
6.3.3.16	operator>=	22
6.3.3.17	operator>=	22
6.3.3.18	operator>=	23
7	Namespace Documentation	25
7.1	jeod Namespace Reference	25
7.1.1	Detailed Description	26
8	Data Structure Documentation	27
8.1	jeod::JeodAssociativeContainer< ElemType, ContainerType > Class Template Reference	27
8.1.1	Detailed Description	29
8.1.2	Member Typedef Documentation	29
8.1.2.1	base_container_type	29
8.1.2.2	key_compare	29
8.1.2.3	key_type	29
8.1.2.4	this_container_type	29
8.1.2.5	value_compare	29
8.1.3	Constructor & Destructor Documentation	30
8.1.3.1	~JeodAssociativeContainer	30
8.1.3.2	JeodAssociativeContainer	30
8.1.3.3	JeodAssociativeContainer	30
8.1.3.4	JeodAssociativeContainer	30
8.1.4	Member Function Documentation	30
8.1.4.1	count	30
8.1.4.2	equal_range	30
8.1.4.3	equal_range	31
8.1.4.4	erase	31
8.1.4.5	erase	31
8.1.4.6	erase	31
8.1.4.7	find	31
8.1.4.8	find	31
8.1.4.9	insert	32
8.1.4.10	insert	32
8.1.4.11	key_comp	32

8.1.4.12	lower_bound	32
8.1.4.13	lower_bound	32
8.1.4.14	upper_bound	32
8.1.4.15	upper_bound	33
8.1.4.16	value_comp	33
8.2	jeod::JeodCheckpointable Class Reference	33
8.2.1	Detailed Description	34
8.2.2	Constructor & Destructor Documentation	34
8.2.2.1	JeodCheckpointable	34
8.2.2.2	~JeodCheckpointable	35
8.2.2.3	JeodCheckpointable	35
8.2.3	Member Function Documentation	35
8.2.3.1	advance_checkpoint	35
8.2.3.2	get_final_name	35
8.2.3.3	get_final_value	35
8.2.3.4	get_init_name	35
8.2.3.5	get_init_value	35
8.2.3.6	get_item_name	36
8.2.3.7	get_item_value	36
8.2.3.8	initialize_checkpointable	36
8.2.3.9	is_checkpoint_finished	36
8.2.3.10	operator=	36
8.2.3.11	perform_restore_action	37
8.2.3.12	post_checkpoint	37
8.2.3.13	post_restart	37
8.2.3.14	pre_checkpoint	37
8.2.3.15	pre_restart	37
8.2.3.16	start_checkpoint	38
8.2.3.17	undo_initialize_checkpointable	38
8.2.4	Friends And Related Function Documentation	38
8.2.4.1	init_attrjeod__JeodCheckpointable	38
8.2.4.2	InputProcessor	38
8.3	jeod::JeodContainer< ContainerType, ElemType > Class Template Reference	38
8.3.1	Detailed Description	40
8.3.2	Member Typedef Documentation	40
8.3.2.1	stl_container_type	40
8.3.2.2	this_container_type	40
8.3.3	Constructor & Destructor Documentation	40
8.3.3.1	JeodContainer	40
8.3.3.2	JeodContainer	40

8.3.3.3	JeodContainer	41
8.3.3.4	~JeodContainer	41
8.3.4	Member Function Documentation	41
8.3.4.1	advance_checkpoint	41
8.3.4.2	get_final_name	41
8.3.4.3	get_init_name	41
8.3.4.4	get_item_name	42
8.3.4.5	initialize_checkpointable	42
8.3.4.6	is_checkpoint_finished	42
8.3.4.7	operator=	42
8.3.4.8	operator=	42
8.3.4.9	perform_cleanup_action	43
8.3.4.10	perform_insert_action	43
8.3.4.11	perform_restore_action	43
8.3.4.12	start_checkpoint	44
8.3.4.13	swap_contents	44
8.3.4.14	swap_contents	44
8.3.5	Friends And Related Function Documentation	44
8.3.5.1	init_attrjeod__JeodContainer	44
8.3.5.2	InputProcessor	44
8.3.6	Field Documentation	44
8.3.6.1	checkpoint_iter	44
8.3.6.2	elem_type_descriptor	44
8.4	jeod::JeodList< ElemType > Class Template Reference	45
8.4.1	Detailed Description	46
8.4.2	Member Typedef Documentation	47
8.4.2.1	jeod_sequence_container_type	47
8.4.2.2	jeod_stl_container_type	47
8.4.2.3	stl_container_type	47
8.4.2.4	this_container_type	47
8.4.3	Constructor & Destructor Documentation	47
8.4.3.1	~JeodList	47
8.4.3.2	JeodList	47
8.4.3.3	JeodList	47
8.4.3.4	JeodList	48
8.4.4	Member Function Documentation	49
8.4.4.1	merge	49
8.4.4.2	merge	49
8.4.4.3	operator=	49
8.4.4.4	operator=	49

8.4.4.5	pop_front	49
8.4.4.6	push_front	50
8.4.4.7	remove	50
8.4.4.8	remove_if	50
8.4.4.9	reverse	50
8.4.4.10	sort	50
8.4.4.11	sort	50
8.4.4.12	splice	51
8.4.4.13	splice	51
8.4.4.14	splice	51
8.4.4.15	unique	51
8.4.4.16	unique	51
8.5	jeod::JeodObjectContainer< ContainerType, ElemType > Class Template Reference	52
8.5.1	Detailed Description	53
8.5.2	Constructor & Destructor Documentation	53
8.5.2.1	JeodObjectContainer	53
8.5.2.2	JeodObjectContainer	53
8.5.2.3	JeodObjectContainer	53
8.5.2.4	~JeodObjectContainer	54
8.5.3	Member Function Documentation	54
8.5.3.1	advance_checkpoint	54
8.5.3.2	get_final_value	54
8.5.3.3	get_item_value	54
8.5.3.4	operator=	55
8.5.3.5	operator=	55
8.5.3.6	perform_cleanup_action	55
8.5.3.7	perform_insert_action	55
8.5.3.8	post_checkpoint	56
8.5.3.9	post_restart	56
8.5.3.10	pre_checkpoint	56
8.5.3.11	start_checkpoint	56
8.5.4	Friends And Related Function Documentation	56
8.5.4.1	init_attrjeod__JeodObjectContainer	56
8.5.4.2	InputProcessor	56
8.5.5	Field Documentation	57
8.5.5.1	copy	57
8.5.5.2	index	57
8.6	jeod::JeodObjectList< ElemType > Class Template Reference	57
8.6.1	Detailed Description	57
8.6.2	Member Typedef Documentation	57

8.6.2.1	type	58
8.7	jeod::JeodObjectSet< ElemType > Class Template Reference	58
8.7.1	Detailed Description	58
8.7.2	Member Typedef Documentation	58
8.7.2.1	type	58
8.8	jeod::JeodObjectVector< ElemType > Class Template Reference	58
8.8.1	Detailed Description	59
8.8.2	Member Typedef Documentation	59
8.8.2.1	type	59
8.9	jeod::JeodPointerContainer< ContainerType, ElemType > Class Template Reference	59
8.9.1	Detailed Description	60
8.9.2	Constructor & Destructor Documentation	60
8.9.2.1	JeodPointerContainer	60
8.9.2.2	JeodPointerContainer	60
8.9.2.3	JeodPointerContainer	61
8.9.2.4	~JeodPointerContainer	61
8.9.3	Member Function Documentation	61
8.9.3.1	get_item_value	61
8.9.3.2	initialize_checkpointable	61
8.9.3.3	operator=	62
8.9.3.4	operator=	62
8.9.3.5	perform_insert_action	62
8.9.4	Field Documentation	62
8.9.4.1	base_type_descriptor	62
8.10	jeod::JeodPointerList< ElemType > Class Template Reference	63
8.10.1	Detailed Description	63
8.10.2	Member Typedef Documentation	63
8.10.2.1	type	63
8.11	jeod::JeodPointerSet< ElemType > Class Template Reference	63
8.11.1	Detailed Description	63
8.11.2	Member Typedef Documentation	64
8.11.2.1	type	64
8.12	jeod::JeodPointerVector< ElemType > Class Template Reference	64
8.12.1	Detailed Description	64
8.12.2	Member Typedef Documentation	64
8.12.2.1	type	64
8.13	jeod::JeodPrimitiveContainer< ContainerType, ElemType > Class Template Reference	65
8.13.1	Detailed Description	65
8.13.2	Constructor & Destructor Documentation	66
8.13.2.1	JeodPrimitiveContainer	66

8.13.2.2	JeodPrimitiveContainer	66
8.13.2.3	JeodPrimitiveContainer	66
8.13.2.4	~JeodPrimitiveContainer	66
8.13.3	Member Function Documentation	66
8.13.3.1	get_item_value	66
8.13.3.2	operator=	67
8.13.3.3	operator=	67
8.13.3.4	perform_insert_action	67
8.13.4	Field Documentation	67
8.13.4.1	serializer	67
8.14	jeod::JeodPrimitiveList< ElemType > Class Template Reference	68
8.14.1	Detailed Description	68
8.14.2	Member Typedef Documentation	68
8.14.2.1	type	68
8.15	jeod::JeodPrimitiveSerializer< Type > Class Template Reference	68
8.15.1	Detailed Description	69
8.15.2	Constructor & Destructor Documentation	70
8.15.2.1	JeodPrimitiveSerializer	70
8.15.2.2	~JeodPrimitiveSerializer	70
8.15.2.3	JeodPrimitiveSerializer	70
8.15.3	Member Function Documentation	70
8.15.3.1	from_string	70
8.15.3.2	from_string	70
8.15.3.3	from_string	70
8.15.3.4	from_string	70
8.15.3.5	from_string	70
8.15.3.6	operator=	71
8.15.3.7	to_string	71
8.15.3.8	to_string	71
8.15.3.9	to_string	71
8.15.3.10	to_string	71
8.15.3.11	to_string	71
8.16	jeod::JeodPrimitiveSerializerBase Class Reference	71
8.16.1	Detailed Description	72
8.16.2	Constructor & Destructor Documentation	72
8.16.2.1	JeodPrimitiveSerializerBase	72
8.16.2.2	~JeodPrimitiveSerializerBase	72
8.16.3	Member Function Documentation	72
8.16.3.1	deserialize_double	72
8.16.3.2	deserialize_float	73

8.16.3.3	deserialize_long_double	73
8.16.3.4	deserialize_string	73
8.16.3.5	serialize_double	73
8.16.3.6	serialize_float	74
8.16.3.7	serialize_long_double	74
8.16.3.8	serialize_string	74
8.17	jeod::JeodPrimitiveSet< ElemType > Class Template Reference	75
8.17.1	Detailed Description	75
8.17.2	Member Typedef Documentation	75
8.17.2.1	type	75
8.18	jeod::JeodPrimitiveVector< ElemType > Class Template Reference	75
8.18.1	Detailed Description	76
8.18.2	Member Typedef Documentation	76
8.18.2.1	type	76
8.19	jeod::JeodSequenceContainer< ElemType, ContainerType > Class Template Reference	76
8.19.1	Detailed Description	77
8.19.2	Member Typedef Documentation	78
8.19.2.1	base_container_type	78
8.19.2.2	this_container_type	78
8.19.3	Constructor & Destructor Documentation	78
8.19.3.1	~JeodSequenceContainer	78
8.19.3.2	JeodSequenceContainer	78
8.19.3.3	JeodSequenceContainer	78
8.19.3.4	JeodSequenceContainer	79
8.19.4	Member Function Documentation	80
8.19.4.1	assign	80
8.19.4.2	assign	80
8.19.4.3	back	80
8.19.4.4	back	80
8.19.4.5	erase	80
8.19.4.6	erase	81
8.19.4.7	front	81
8.19.4.8	front	81
8.19.4.9	insert	81
8.19.4.10	insert	81
8.19.4.11	pop_back	82
8.19.4.12	push_back	82
8.19.4.13	resize	82
8.20	jeod::JeodSet< ElemType > Class Template Reference	82
8.20.1	Detailed Description	83

8.20.2	Member Typedef Documentation	83
8.20.2.1	jeod_associative_container_type	83
8.20.2.2	jeod_stl_container_type	84
8.20.2.3	stl_container_type	84
8.20.2.4	this_container_type	84
8.20.3	Constructor & Destructor Documentation	84
8.20.3.1	~JeodSet	84
8.20.3.2	JeodSet	84
8.20.3.3	JeodSet	84
8.20.3.4	JeodSet	84
8.20.4	Member Function Documentation	85
8.20.4.1	operator=	85
8.20.4.2	operator=	85
8.21	jeod::JeodSTLContainer< ElemType, ContainerType > Class Template Reference	85
8.21.1	Detailed Description	87
8.21.2	Member Typedef Documentation	87
8.21.2.1	allocator_type	87
8.21.2.2	const_iterator	88
8.21.2.3	const_reference	88
8.21.2.4	const_reverse_iterator	88
8.21.2.5	difference_type	88
8.21.2.6	iterator	88
8.21.2.7	reference	88
8.21.2.8	reverse_iterator	88
8.21.2.9	size_type	88
8.21.2.10	this_container_type	89
8.21.2.11	value_type	89
8.21.3	Constructor & Destructor Documentation	89
8.21.3.1	~JeodSTLContainer	89
8.21.3.2	JeodSTLContainer	89
8.21.3.3	JeodSTLContainer	89
8.21.3.4	JeodSTLContainer	89
8.21.4	Member Function Documentation	89
8.21.4.1	begin	90
8.21.4.2	begin	90
8.21.4.3	clear	90
8.21.4.4	empty	90
8.21.4.5	end	90
8.21.4.6	end	90
8.21.4.7	get_allocator	90

8.21.4.8	insert	90
8.21.4.9	max_size	91
8.21.4.10	operator const ContainerType &	91
8.21.4.11	operator ContainerType &	91
8.21.4.12	operator=	91
8.21.4.13	operator=	91
8.21.4.14	rbegin	91
8.21.4.15	rbegin	92
8.21.4.16	rend	92
8.21.4.17	rend	92
8.21.4.18	size	92
8.21.4.19	swap	92
8.21.4.20	swap	92
8.21.5	Field Documentation	92
8.21.5.1	contents	93
8.22	jeod::JeodVector< ElemType > Class Template Reference	93
8.22.1	Detailed Description	94
8.22.2	Member Typedef Documentation	95
8.22.2.1	jeod_sequence_container_type	95
8.22.2.2	jeod_stl_container_type	95
8.22.2.3	stl_container_type	95
8.22.2.4	this_container_type	95
8.22.3	Constructor & Destructor Documentation	95
8.22.3.1	~JeodVector	95
8.22.3.2	JeodVector	95
8.22.3.3	JeodVector	95
8.22.3.4	JeodVector	96
8.22.4	Member Function Documentation	97
8.22.4.1	at	97
8.22.4.2	at	97
8.22.4.3	capacity	97
8.22.4.4	operator=	97
8.22.4.5	operator=	97
8.22.4.6	operator[]	98
8.22.4.7	operator[]	98
8.22.4.8	reserve	98
8.23	jeod::SimpleCheckpointable Class Reference	98
8.23.1	Detailed Description	99
8.23.2	Constructor & Destructor Documentation	99
8.23.2.1	SimpleCheckpointable	99

8.23.2.2	~SimpleCheckpointable	100
8.23.2.3	SimpleCheckpointable	100
8.23.3	Member Function Documentation	100
8.23.3.1	advance_checkpoint	100
8.23.3.2	get_init_name	100
8.23.3.3	get_item_name	100
8.23.3.4	get_item_value	100
8.23.3.5	is_checkpoint_finished	100
8.23.3.6	operator=	101
8.23.3.7	perform_restore_action	101
8.23.3.8	simple_restore	101
8.23.3.9	start_checkpoint	101
8.23.4	Friends And Related Function Documentation	101
8.23.4.1	init_attrjeod__SimpleCheckpointable	101
8.23.4.2	InputProcessor	101
9	File Documentation	103
9.1	checkpointable.hh File Reference	103
9.1.1	Detailed Description	103
9.2	container.hh File Reference	103
9.2.1	Detailed Description	104
9.3	jeod_associative_container.hh File Reference	104
9.3.1	Detailed Description	104
9.4	jeod_container_compare.hh File Reference	104
9.4.1	Detailed Description	106
9.5	jeod_list.hh File Reference	106
9.5.1	Detailed Description	106
9.6	jeod_sequence_container.hh File Reference	107
9.6.1	Detailed Description	107
9.7	jeod_set.hh File Reference	107
9.7.1	Detailed Description	107
9.8	jeod_stl_container.hh File Reference	108
9.8.1	Detailed Description	108
9.9	jeod_vector.hh File Reference	108
9.9.1	Detailed Description	108
9.10	object_container.hh File Reference	109
9.10.1	Detailed Description	109
9.10.2	Macro Definition Documentation	109
9.10.2.1	JEOD_OBJECT_CONTAINER	109
9.11	object_list.hh File Reference	109

9.11.1 Detailed Description	110
9.12 object_set.hh File Reference	110
9.12.1 Detailed Description	110
9.13 object_vector.hh File Reference	110
9.13.1 Detailed Description	111
9.14 pointer_container.hh File Reference	111
9.14.1 Detailed Description	111
9.14.2 Macro Definition Documentation	112
9.14.2.1 JEOD_POINTER_CONTAINER	112
9.15 pointer_list.hh File Reference	112
9.15.1 Detailed Description	112
9.16 pointer_set.hh File Reference	112
9.16.1 Detailed Description	113
9.17 pointer_vector.hh File Reference	113
9.17.1 Detailed Description	113
9.18 primitive_container.hh File Reference	113
9.18.1 Detailed Description	114
9.18.2 Macro Definition Documentation	114
9.18.2.1 JEOD_PRIMITIVE_CONTAINER	114
9.19 primitive_list.hh File Reference	114
9.19.1 Detailed Description	114
9.20 primitive_serializer.cc File Reference	114
9.20.1 Detailed Description	115
9.21 primitive_serializer.hh File Reference	115
9.21.1 Detailed Description	115
9.22 primitive_set.hh File Reference	115
9.22.1 Detailed Description	116
9.23 primitive_vector.hh File Reference	116
9.23.1 Detailed Description	116
9.24 simple_checkpointable.hh File Reference	116
9.24.1 Detailed Description	117

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 >	38
jeod::JeodObjectContainer< ContainerType, ElemType >	52
jeod::JeodPrimitiveContainer< ContainerType, ElemType >	65
jeod::JeodContainer< ContainerType, ElemType * >	38
jeod::JeodPointerContainer< ContainerType, ElemType >	59
jeod::JeodCheckpointable	33
jeod::JeodContainer< ContainerType, ElemType >	38
jeod::SimpleCheckpointable	98
jeod::JeodContainer< ContainerType, ElemType * >	38
jeod::JeodObjectList< ElemType >	57
jeod::JeodObjectSet< ElemType >	58
jeod::JeodObjectVector< ElemType >	58
jeod::JeodPointerList< ElemType >	63
jeod::JeodPointerSet< ElemType >	63
jeod::JeodPointerVector< ElemType >	64
jeod::JeodPrimitiveList< ElemType >	68
jeod::JeodPrimitiveSerializerBase	71
jeod::JeodPrimitiveSerializer< Type >	68
jeod::JeodPrimitiveSerializer< ElemType >	68
jeod::JeodPrimitiveSet< ElemType >	75
jeod::JeodPrimitiveVector< ElemType >	75
jeod::JeodSTLContainer< ElemType, ContainerType >	85
jeod::JeodAssociativeContainer< ElemType, ContainerType >	27
jeod::JeodSequenceContainer< ElemType, ContainerType >	76
jeod::JeodSTLContainer< ElemType, std::list< ElemType > >	85
jeod::JeodSequenceContainer< ElemType, std::list< ElemType > >	76
jeod::JeodList< ElemType >	45
jeod::JeodSTLContainer< ElemType, std::set< ElemType > >	85
jeod::JeodAssociativeContainer< ElemType, std::set< ElemType > >	27
jeod::JeodSet< ElemType >	82
jeod::JeodSTLContainer< ElemType, std::vector< ElemType > >	85
jeod::JeodSequenceContainer< ElemType, std::vector< ElemType > >	76
jeod::JeodVector< ElemType >	93

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	33
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 >	38
A JeodContainer is a JEOD STL sequence container replacement whose contents are checkpointable and restorable	
jeod::JeodList< ElemType >	45
The JEOD replacement for std::list	
jeod::JeodObjectContainer< ContainerType, ElemType >	52
A JeodObjectContainer is a JeodContainer that contains objects of type ElemType	
jeod::JeodObjectList< ElemType >	57
Defines a registry for defining a checkpointable list of objects	
jeod::JeodObjectSet< ElemType >	58
Defines a registry for defining a checkpointable set of objects	
jeod::JeodObjectVector< ElemType >	58
Defines a registry for defining a checkpointable vector of objects	
jeod::JeodPointerContainer< ContainerType, ElemType >	59
A JeodPointerContainer is a JeodContainer that contains pointers to objects of type ElemType	
jeod::JeodPointerList< ElemType >	63
Defines a registry for defining a checkpointable list of pointers	
jeod::JeodPointerSet< ElemType >	63
Defines a registry for defining a checkpointable set of pointers	
jeod::JeodPointerVector< ElemType >	64
Defines a registry for defining a checkpointable vector of pointers	
jeod::JeodPrimitiveContainer< ContainerType, ElemType >	65
A JeodPrimitiveContainer is a JeodContainer that contains primitive data of type ElemType	
jeod::JeodPrimitiveList< ElemType >	68
Defines a registry for defining a checkpointable list of primitives	
jeod::JeodPrimitiveSerializer< Type >	68
Serializer / deserializer for primitive data	
jeod::JeodPrimitiveSerializerBase	71
Base class for serializing / deserializing primitive data	
jeod::JeodPrimitiveSet< ElemType >	75
Defines a registry for defining a checkpointable set of primitives	

jeod::JeodPrimitiveVector< ElemType >	
Defines a registry for defining a checkpointable vector of primitives	75
jeod::JeodSequenceContainer< ElemType, ContainerType >	
This is the base class for the JEOD replacements of the STL sequence containers	76
jeod::JeodSet< ElemType >	
The JEOD replacement for std::set	82
jeod::JeodSTLContainer< ElemType, ContainerType >	
This is the base class for the JEOD replacements of the STL containers	85
jeod::JeodVector< ElemType >	
The JEOD replacement for std::vector	93
jeod::SimpleCheckpointable	
Simple checkpoint/restart interface by which an object can complete the restart process	98

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	103
container.hh	Define the class JeodContainer, which adds checkpointability to an STL sequence container replacement	103
jeod_associative_container.hh	Define checkpointable replacements for STL associative containers	104
jeod_container_compare.hh	Define comparison operators for JEOD STL container	104
jeod_list.hh	Define the class template JeodList	106
jeod_sequence_container.hh	Define checkpointable replacements for STL sequence containers	107
jeod_set.hh	Define the class template JeodSet	107
jeod_stl_container.hh	Define checkpointable replacements for STL containers	108
jeod_vector.hh	Define class template JeodVector	108
object_container.hh	Define class template JeodObjectContainer	109
object_list.hh	Define checkpointable replacements for STL sequence containers	109
object_set.hh	Define checkpointable replacements for STL associative containers	110
object_vector.hh	Define checkpointable replacements for STL sequence containers	110
pointer_container.hh	Define class template JeodPointerContainer	111
pointer_list.hh	Define checkpointable replacements for STL sequence containers	112
pointer_set.hh	Define checkpointable replacements for STL associative containers	112
pointer_vector.hh	Define checkpointable replacements for STL sequence containers	113
primitive_container.hh	Define class template JeodPrimitiveContainer	113

primitive_list.hh	
Define checkpointable replacements for STL sequence containers	114
primitive_serializer.cc	
Define class JeodPrimitiveSerializerBase static methods	114
primitive_serializer.hh	
Define class template JeodPrimitiveSerializer	115
primitive_set.hh	
Define checkpointable replacements for STL associative containers	115
primitive_vector.hh	
Define checkpointable replacements for STL sequence containers	116
simple_checkpointable.hh	
Define the class SimpleCheckpointable	116

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::JeodSTL-`
`Container< 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::JeodSTL-`
`Container< 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 #define __USE_ISOC99

Definition at line 26 of file primitive_serializer.cc.

6.3.3 Function Documentation

6.3.3.1 `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

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

Returns

`x != y`

Definition at line 321 of file jeod_container_compare.hh.

6.3.3.2 `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

x	Comparand
y	Comparand

Returns
 $x \neq y$

Definition at line 335 of file jeod_container_compare.hh.

```
6.3.3.3  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

x	Comparand
y	Comparand

Returns
 $x \neq y$

Definition at line 349 of file jeod_container_compare.hh.

```
6.3.3.4  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 149 of file jeod_container_compare.hh.

```
6.3.3.5  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 163 of file jeod_container_compare.hh.

```
6.3.3.6  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

x	Comparand
y	Comparand

Returns
 $x < y$

Definition at line 177 of file jeod_container_compare.hh.

6.3.3.7 `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

x	Comparand
y	Comparand

Returns
 $x \leq y$

Definition at line 364 of file jeod_container_compare.hh.

6.3.3.8 `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

x	Comparand
y	Comparand

Returns
 $x \leq y$

Definition at line 378 of file jeod_container_compare.hh.

6.3.3.9 `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

x	Comparand
y	Comparand

Returns
 $x \leq y$

Definition at line 392 of file jeod_container_compare.hh.


```
6.3.3.10  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 192 of file jeod_container_compare.hh.

6.3.3.11 `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

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

Returns

$x == y$

Definition at line 206 of file jeod_container_compare.hh.

6.3.3.12 `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

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

Returns

$x == y$

Definition at line 220 of file jeod_container_compare.hh.

6.3.3.13 `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 235 of file jeod_container_compare.hh.

```
6.3.3.14  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

x	Comparand
y	Comparand

Returns
 $x > y$

Definition at line 249 of file jeod_container_compare.hh.

```
6.3.3.15  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

x	Comparand
y	Comparand

Returns
 $x > y$

Definition at line 263 of file jeod_container_compare.hh.

```
6.3.3.16  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

x	Comparand
y	Comparand

Returns
 $x >= y$

Definition at line 278 of file jeod_container_compare.hh.

```
6.3.3.17  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

x	Comparand
y	Comparand

Returns
 $x >= y$

Definition at line 292 of file jeod_container_compare.hh.

```
6.3.3.18  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 306 of file jeod_container_compare.hh.

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- 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 [JeodAssociativeContainer](#)
This is the base class for the JEOD replacements of the STL associative containers.
- class [JeodList](#)
The JEOD replacement for `std::list`.
- 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 [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 [JeodPrimitiveSerializerBase](#)
 - Base class for serializing / deserializing primitive data.*
- class [JeodPrimitiveSerializer](#)
 - Serializer / deserializer for 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 [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

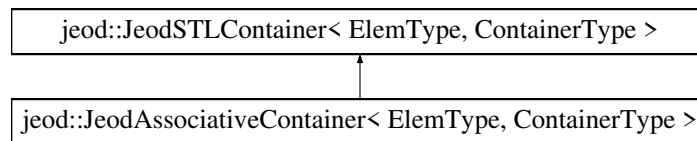
Data Structure Documentation

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

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

Public Member Functions

- virtual [~JeodAssociativeContainer](#) (void)
Destructor.
- [key_compare key_comp](#) (void) const

- Returns the key comparison object used to populate the contents.*
- `value_compare value_comp` (void) 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.

Protected Member Functions

- `JeodAssociativeContainer` (void)
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 69 of file `jeod_associative_container.hh`.

8.1.2 Member Typedef Documentation

8.1.2.1 `template<typename ElemType, typename ContainerType> typedef JeodSTLContainer<ElemType, ContainerType>
jeod::JeodAssociativeContainer< ElemType, ContainerType >::base_container_type`

The [JeodSTLContainer](#).

Definition at line 85 of file `jeod_associative_container.hh`.

8.1.2.2 `template<typename ElemType, typename ContainerType> typedef ContainerType::key_compare
jeod::JeodAssociativeContainer< ElemType, ContainerType >::key_compare`

Import the `ContainerType::key_compare`.

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

8.1.2.3 `template<typename ElemType, typename ContainerType> typedef ContainerType::key_type
jeod::JeodAssociativeContainer< ElemType, ContainerType >::key_type`

Import the `ContainerType::key_type`.

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

8.1.2.4 `template<typename ElemType, typename ContainerType> typedef JeodAssociativeContainer<ElemType,
ContainerType> jeod::JeodAssociativeContainer< ElemType, ContainerType >::this_container_type`

This type.

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

8.1.2.5 `template<typename ElemType, typename ContainerType> typedef ContainerType::value_compare
jeod::JeodAssociativeContainer< ElemType, ContainerType >::value_compare`

Import the `ContainerType::value_compare`.

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

8.1.3 Constructor & Destructor Documentation

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

Destructor.

Definition at line 118 of file jeod_associative_container.hh.

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

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.

Definition at line 314 of file jeod_associative_container.hh.

8.1.3.3 `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 320 of file jeod_associative_container.hh.

8.1.3.4 `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 328 of file jeod_associative_container.hh.

8.1.4 Member Function Documentation

8.1.4.1 `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 154 of file jeod_associative_container.hh.

8.1.4.2 `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 218 of file jeod_associative_container.hh.

8.1.4.3 `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 228 of file jeod_associative_container.hh.

8.1.4.4 `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 271 of file jeod_associative_container.hh.

8.1.4.5 `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 283 of file jeod_associative_container.hh.

8.1.4.6 `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 295 of file jeod_associative_container.hh.

8.1.4.7 `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 163 of file jeod_associative_container.hh.

8.1.4.8 `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 172 of file jeod_associative_container.hh.

```
8.1.4.9  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 248 of file jeod_associative_container.hh.

```
8.1.4.10 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 260 of file jeod_associative_container.hh.

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

Returns the key comparison object used to populate the contents.

Definition at line 127 of file jeod_associative_container.hh.

```
8.1.4.12 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 181 of file jeod_associative_container.hh.

```
8.1.4.13 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 190 of file jeod_associative_container.hh.

```
8.1.4.14 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 199 of file jeod_associative_container.hh.

8.1.4.15 `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 208 of file `jeod_associative_container.hh`.

8.1.4.16 `template<typename ElemType, typename ContainerType> value_compare jeod::Jeod-
AssociativeContainer< ElemType, ContainerType >::value_comp (void) const
[inline]`

Returns the value comparison object used to populate the contents.

Definition at line 136 of file `jeod_associative_container.hh`.

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

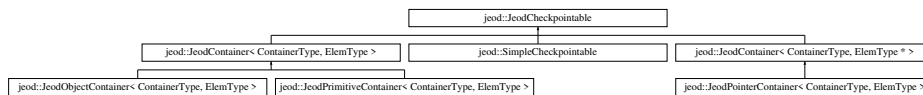
- [jeod_associative_container.hh](#)

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 constructor; does nothing.
- virtual [~JeodCheckpointable](#) ()
Destructor; does nothing.
- virtual void [pre_checkpoint](#) (void)
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](#) (void)
In general, perform object-specific operations that need to be performed after checkpoint completion, typically freeing memory used for checkpointing.
- virtual void [pre_restart](#) (void)
In general, perform object-specific operations that need to be performed in anticipation of a restart, typically releasing resources.
- virtual void [post_restart](#) (void)
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` (void)
In general, return the value of the initialization action.
- virtual const std::string `get_final_name` (void)
In general, return the name of the finalization action.
- virtual const std::string `get_final_value` (void)
In general, return the value of the finalization action.
- virtual void `start_checkpoint` (void)=0
Prepare to checkpoint the object in question.
- virtual void `advance_checkpoint` (void)=0
Advance to the next item to be checkpointed.
- virtual bool `is_checkpoint_finished` (void)=0
Return true if all contents have been checkpointed, false otherwise.
- virtual const std::string `get_init_name` (void)=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` (void)=0
Return the name of the action that will restore the value at the current checkpoint position.
- virtual const std::string `get_item_value` (void)=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.

Private Member Functions

- `JeodCheckpointable` (const `JeodCheckpointable` &)
Not implemented.
- `JeodCheckpointable` & `operator=` (const `JeodCheckpointable` &)
Not implemented.

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 47 of file checkpointable.hh.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 `jeod::JeodCheckpointable::JeodCheckpointable (void) [inline]`

Default constructor; does nothing.

Definition at line 166 of file checkpointable.hh.

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

Destructor; does nothing.

Definition at line 177 of file checkpointable.hh.

8.2.2.3 `jeod::JeodCheckpointable::JeodCheckpointable (const JeodCheckpointable &) [private]`

Not implemented.

8.2.3 Member Function Documentation

8.2.3.1 `virtual void jeod::JeodCheckpointable::advance_checkpoint (void) [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 `const std::string jeod::JeodCheckpointable::get_final_name (void) [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 207 of file checkpointable.hh.

8.2.3.3 `const std::string jeod::JeodCheckpointable::get_final_value (void) [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 222 of file checkpointable.hh.

8.2.3.4 `virtual const std::string jeod::JeodCheckpointable::get_init_name (void) [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 `const std::string jeod::JeodCheckpointable::get_init_value (void) [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 192 of file checkpointable.hh.

8.2.3.6 `virtual const std::string jeod::JeodCheckpointable::get_item_name (void) [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 `virtual const std::string jeod::JeodCheckpointable::get_item_value (void) [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 `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 300 of file checkpointable.hh.

8.2.3.9 `virtual bool jeod::JeodCheckpointable::is_checkpoint_finished (void) [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 `JeodCheckpointable& jeod::JeodCheckpointable::operator= (const JeodCheckpointable &) [private]`

Not implemented.

8.2.3.11 `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 `void jeod::JeodCheckpointable::post_checkpoint (void) [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 252 of file checkpointable.hh.

8.2.3.13 `void jeod::JeodCheckpointable::post_restart (void) [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 281 of file checkpointable.hh.

8.2.3.14 `void jeod::JeodCheckpointable::pre_checkpoint (void) [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 237 of file checkpointable.hh.

8.2.3.15 `void jeod::JeodCheckpointable::pre_restart (void) [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 267 of file checkpointable.hh.

8.2.3.16 `virtual void jeod::JeodCheckpointable::start_checkpoint (void) [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 `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 319 of file checkpointable.hh.

8.2.4 Friends And Related Function Documentation

8.2.4.1 `void init_attrjeod__JeodCheckpointable () [friend]`

8.2.4.2 `friend class InputProcessor [friend]`

Definition at line 48 of file checkpointable.hh.

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

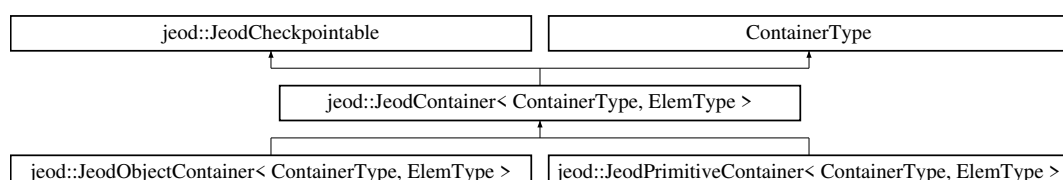
- [checkpointable.hh](#)

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

- typedef [JeodContainer](#)
< ContainerType, ElemType > [this_container_type](#)

This particular *JeodContainer* type.

- typedef
ContainerType::stl_container_type *stl_container_type*
Import the ContainerType's container type.

Public Member Functions

- *JeodContainer* (void)
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.
- virtual ~*JeodContainer* (void)
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.
- virtual void *initialize_checkpointable* (const void *container, const std::type_info &container_type, const std::string elem_name)
Initialize a checkpointable object, called by the checkpoint manager.
- virtual void *start_checkpoint* (void)
Prepare to checkpoint the object.
- virtual void *advance_checkpoint* (void)
Advance to the next item to be checkpointed.
- virtual bool *is_checkpoint_finished* (void)
Indicate whether the checkpoint dump of this object is finished.
- virtual const std::string *get_init_name* (void)
Names the action to be performed prior to performing any of the restore actions.
- virtual const std::string *get_item_name* (void)
Return the name of the action to be printed along with the current value.
- virtual const std::string *get_final_name* (void)
Names the action to be performed after to performing any of the restore actions.
- virtual int *perform_restore_action* (const std::string &action_name, const std::string &action_value)
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 49 of file container.hh.

8.3.2 Member Typedef Documentation

8.3.2.1 `template<typename ContainerType, typename ElemType> typedef ContainerType::stl_container_type
jeod::JeodContainer< ContainerType, ElemType >::stl_container_type`

Import the ContainerType's container type.

Definition at line 68 of file container.hh.

8.3.2.2 `template<typename ContainerType, typename ElemType> typedef JeodContainer<ContainerType, ElemType>
jeod::JeodContainer< ContainerType, ElemType >::this_container_type`

This particular [JeodContainer](#) type.

Definition at line 62 of file container.hh.

8.3.3 Constructor & Destructor Documentation

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

Default constructor.

Definition at line 76 of file container.hh.

8.3.3.2 `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

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

Definition at line 92 of file container.hh.

8.3.3.3 `template<typename ContainerType, typename ElemType> jeod::JeodContainer< ContainerType, ElemType >::JeodContainer (const std::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 108 of file container.hh.

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

Destructor.

Definition at line 149 of file container.hh.

8.3.4 Member Function Documentation

8.3.4.1 `template<typename ContainerType, typename ElemType> virtual void jeod::JeodContainer< ContainerType, ElemType >::advance_checkpoint (void) [inline],[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 234 of file container.hh.

Referenced by [jeod::JeodObjectContainer< ContainerType, ElemType >::advance_checkpoint\(\)](#).

8.3.4.2 `template<typename ContainerType, typename ElemType> virtual const std::string jeod::JeodContainer< ContainerType, ElemType >::get_final_name (void) [inline],[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 277 of file container.hh.

8.3.4.3 `template<typename ContainerType, typename ElemType> virtual const std::string jeod::JeodContainer< ContainerType, ElemType >::get_init_name (void) [inline],[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 256 of file container.hh.

8.3.4.4 `template<typename ContainerType, typename ElemType> virtual const std::string jeod::JeodContainer< ContainerType, ElemType >::get_item_name (void) [inline], [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 266 of file container.hh.

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

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

Definition at line 204 of file container.hh.

Referenced by [jeod::JeodPointerContainer< ContainerType, ElemType >::initialize_checkpointable\(\)](#).

8.3.4.6 `template<typename ContainerType, typename ElemType> virtual bool jeod::JeodContainer< ContainerType, ElemType >::is_checkpoint_finished (void) [inline], [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 245 of file container.hh.

8.3.4.7 `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 124 of file container.hh.

Referenced by [jeod::JeodPrimitiveContainer< ContainerType, ElemType >::operator=\(\)](#), [jeod::JeodPointerContainer< ContainerType, ElemType >::operator=\(\)](#), and [jeod::JeodObjectContainer< ContainerType, ElemType >::operator=\(\)](#).

8.3.4.8 `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 140 of file container.hh.

8.3.4.9 `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 193 of file container.hh.

Referenced by `jeod::JeodContainer< ContainerType, ElemType * >::perform_restore_action()`.

8.3.4.10 `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 >](#).

Referenced by `jeod::JeodContainer< ContainerType, ElemType * >::perform_restore_action()`.

8.3.4.11 `template<typename ContainerType, typename ElemType> virtual int jeod::JeodContainer< ContainerType, ElemType >::perform_restore_action (const std::string & action_name, const std::string & action_value) [inline],[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 293 of file container.hh.

8.3.4.12 `template<typename ContainerType, typename ElemType> virtual void jeod::JeodContainer< ContainerType, ElemType >::start_checkpoint (void) [inline],[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 222 of file `container.hh`.

Referenced by `jeod::JeodObjectContainer< ContainerType, ElemType >::start_checkpoint()`.

8.3.4.13 `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 156 of file `container.hh`.

8.3.4.14 `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 166 of file `container.hh`.

8.3.5 Friends And Related Function Documentation

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

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

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

8.3.6 Field Documentation

8.3.6.1 `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 331 of file `container.hh`.

Referenced by `jeod::JeodContainer< ContainerType, ElemType * >::advance_checkpoint()`, `jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value()`, `jeod::JeodContainer< ContainerType, ElemType * >::is_checkpoint_finished()`, and `jeod::JeodContainer< ContainerType, ElemType * >::start_checkpoint()`.

8.3.6.2 `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 336 of file container.hh.

Referenced by `jeod::JeodObjectContainer< ContainerType, ElemType >::get_final_value()`, `jeod::JeodObjectContainer< ContainerType, ElemType >::get_item_value()`, and `jeod::JeodContainer< ContainerType, ElemType * >::initialize_checkpointable()`.

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

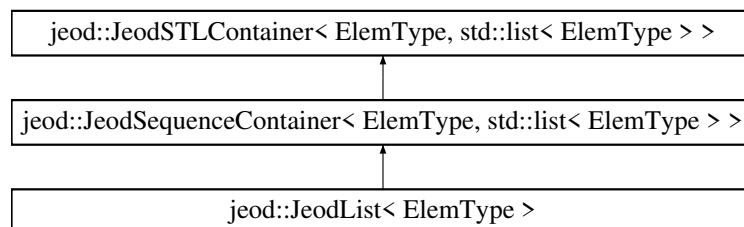
- [container.hh](#)

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

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

Public Member Functions

- virtual `~JeodList` (void)
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](#) (void)
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](#) (void)
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](#) (void)
Sort using the default comparison operator.
- template<typename Compare >
void [sort](#) (Compare comp)
Sort using the provided comparator.
- void [unique](#) (void)
Remove duplicates using the default equality operator.
- template<typename BinaryPredicate >
void [unique](#) (BinaryPredicate comp)
Remove duplicates using the provided comparator.

Protected Member Functions

- [JeodList](#) (void)
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 59 of file jeod_list.hh.

8.4.2 Member Typedef Documentation

8.4.2.1 `template<typename ElemType > typedef JeodSequenceContainer< ElemType, std::list<ElemType> > jeod::JeodList< ElemType >::jeod_sequence_container_type`

The [JeodSequenceContainer](#) type.

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

8.4.2.2 `template<typename ElemType > typedef JeodSTLContainer<ElemType, std::list<ElemType> > jeod::JeodList< ElemType >::jeod_stl_container_type`

The [JeodSTLContainer](#) type.

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

8.4.2.3 `template<typename ElemType > typedef std::list<ElemType> jeod::JeodList< ElemType >::stl_container_type`

The `std::list` itself.

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

8.4.2.4 `template<typename ElemType > typedef JeodList<ElemType> jeod::JeodList< ElemType >::this_container_type`

This particular [JeodList](#) type.

Definition at line 69 of file `jeod_list.hh`.

8.4.3 Constructor & Destructor Documentation

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

Destructor.

Definition at line 97 of file `jeod_list.hh`.

8.4.3.2 `template<typename ElemType > jeod::JeodList< ElemType >::JeodList (void) [inline], [protected]`

Default constructor.

Definition at line 283 of file `jeod_list.hh`.

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

Copy constructor.

Definition at line 288 of file `jeod_list.hh`.

8.4.3.4 `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 296 of file jeod_list.hh.

8.4.4 Member Function Documentation

8.4.4.1 `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 131 of file jeod_list.hh.

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

8.4.4.2 `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 144 of file jeod_list.hh.

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

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

Copy contents from the given source.

Definition at line 106 of file jeod_list.hh.

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

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

Copy contents from the given source.

Definition at line 116 of file jeod_list.hh.

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

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

Deletes the element at the head of the list.

Definition at line 163 of file jeod_list.hh.

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

8.4.4.6 `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 154 of file jeod_list.hh.

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

8.4.4.7 `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 172 of file jeod_list.hh.

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

8.4.4.8 `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 184 of file jeod_list.hh.

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

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

Reverse the list.

Definition at line 193 of file jeod_list.hh.

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

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

Sort using the default comparison operator.

Definition at line 240 of file jeod_list.hh.

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

8.4.4.11 `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 252 of file jeod_list.hh.

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

8.4.4.12 `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 202 of file jeod_list.hh.

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

8.4.4.13 `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 214 of file jeod_list.hh.

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

8.4.4.14 `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 227 of file jeod_list.hh.

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

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

Remove duplicates using the default equality operator.

Definition at line 261 of file jeod_list.hh.

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

8.4.4.16 `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 273 of file jeod_list.hh.

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

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

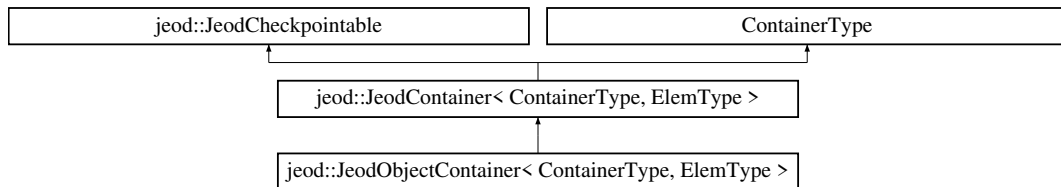
- [jeod_list.hh](#)

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](#) (void)
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](#) (void)
Destruct a [JeodObjectContainer](#).
- virtual void [pre_checkpoint](#) (void)
Prepare to checkpoint a [JeodObjectContainer](#).
- virtual void [post_checkpoint](#) (void)
Cleanup after performing a checkpoint.
- virtual void [post_restart](#) (void)
Cleanup after performing a restart.
- virtual void [start_checkpoint](#) (void)
Prepare to checkpoint the object in question.
- virtual void [advance_checkpoint](#) (void)
Advance to the next item to be checkpointed.
- virtual const std::string [get_item_value](#) (void)
Return the value of the item to be written to the checkpoint file.
- virtual void [perform_insert_action](#) (const std::string &value)
Interpret the provided value and add it to the list.
- virtual const std::string [get_final_value](#) (void)
Return the value of the item to be written to the checkpoint file.
- virtual void [perform_cleanup_action](#) (const std::string &value)
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 50 of file `object_container.hh`.

8.5.2 Constructor & Destructor Documentation

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

Construct a [JeodObjectContainer](#).

Definition at line 58 of file `object_container.hh`.

8.5.2.2 `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 70 of file `object_container.hh`.

8.5.2.3 `template<typename ContainerType , typename ElemType > jeod::JeodObjectContainer< ContainerType, ElemType >::JeodObjectContainer (const typename ContainerType::stl_container_type & 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 83 of file `object_container.hh`.

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

Destruct a [JeodObjectContainer](#).

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

References `jeod::JeodObjectContainer< ContainerType, ElemType >::post_checkpoint()`.

8.5.3 Member Function Documentation

8.5.3.1 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodObjectContainer< ContainerType, ElemType >::advance_checkpoint (void) [inline],[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 180 of file `object_container.hh`.

References `jeod::JeodContainer< ContainerType, ElemType >::advance_checkpoint()`, and `jeod::JeodObjectContainer< ContainerType, ElemType >::index`.

8.5.3.2 `template<typename ContainerType , typename ElemType > virtual const std::string jeod::JeodObjectContainer< ContainerType, ElemType >::get_final_value (void) [inline],[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 218 of file `object_container.hh`.

References `jeod::JeodObjectContainer< ContainerType, ElemType >::copy`, and `jeod::JeodContainer< ContainerType, ElemType >::elem_type_descriptor`.

8.5.3.3 `template<typename ContainerType , typename ElemType > virtual const std::string jeod::JeodObjectContainer< ContainerType, ElemType >::get_item_value (void) [inline],[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 191 of file `object_container.hh`.

References `jeod::JeodObjectContainer< ContainerType, ElemType >::copy`, `jeod::JeodContainer< ContainerType, ElemType >::elem_type_descriptor`, and `jeod::JeodObjectContainer< ContainerType, ElemType >::index`.

8.5.3.4 `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 98 of file object_container.hh.

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

8.5.3.5 `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 111 of file object_container.hh.

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

8.5.3.6 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodObjectContainer< ContainerType, ElemType >::perform_cleanup_action (const std::string & value) [inline],[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 230 of file object_container.hh.

8.5.3.7 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodObjectContainer< ContainerType, ElemType >::perform_insert_action (const std::string & value) [inline],[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 203 of file object_container.hh.

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

Cleanup after performing a checkpoint.

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

Definition at line 148 of file `object_container.hh`.

References `jeod::JeodObjectContainer< ContainerType, ElemType >::copy`.

Referenced by `jeod::JeodObjectContainer< ContainerType, ElemType >::post_restart()`, and `jeod::JeodObjectContainer< ContainerType, ElemType >::~~JeodObjectContainer()`.

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

Cleanup after performing a restart.

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

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

References `jeod::JeodObjectContainer< ContainerType, ElemType >::post_checkpoint()`.

8.5.3.10 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodObjectContainer< ContainerType, ElemType >::pre_checkpoint (void) [inline],[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 131 of file `object_container.hh`.

References `jeod::JeodObjectContainer< ContainerType, ElemType >::copy`.

8.5.3.11 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodObjectContainer< ContainerType, ElemType >::start_checkpoint (void) [inline],[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 169 of file `object_container.hh`.

References `jeod::JeodObjectContainer< ContainerType, ElemType >::index`, and `jeod::JeodContainer< ContainerType, ElemType >::start_checkpoint()`.

8.5.4 Friends And Related Function Documentation

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

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

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

8.5.5 Field Documentation

8.5.5.1 `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 252 of file `object_container.hh`.

Referenced by `jeod::JeodObjectContainer< ContainerType, ElemType >::get_final_value()`, `jeod::JeodObjectContainer< ContainerType, ElemType >::get_item_value()`, `jeod::JeodObjectContainer< ContainerType, ElemType >::post_checkpoint()`, and `jeod::JeodObjectContainer< ContainerType, ElemType >::pre_checkpoint()`.

8.5.5.2 `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 247 of file `object_container.hh`.

Referenced by `jeod::JeodObjectContainer< ContainerType, ElemType >::advance_checkpoint()`, `jeod::JeodObjectContainer< ContainerType, ElemType >::get_item_value()`, and `jeod::JeodObjectContainer< ContainerType, ElemType >::start_checkpoint()`.

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

- typedef [JeodObjectContainer](#)
< [JeodList](#)< ElemType >
, ElemType > [type](#)

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 45 of file `object_list.hh`.

8.6.2 Member Typedef Documentation

```
8.6.2.1 template<typename ElemType > typedef JeodObjectContainer<JeodList<ElemType>, ElemType>
        jeod::JeodObjectList< ElemType >::type
```

Template typedef for a checkpointable list of objects.

Definition at line 50 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

- typedef [JeodObjectContainer](#)
 < [JeodSet](#)< ElemType >
 , ElemType > [type](#)

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 45 of file object_set.hh.

8.7.2 Member Typedef Documentation

```
8.7.2.1 template<typename ElemType > typedef JeodObjectContainer<JeodSet<ElemType>, ElemType>
        jeod::JeodObjectSet< ElemType >::type
```

Template typedef for a checkpointable set of objects.

Definition at line 50 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

- typedef [JeodObjectContainer](#)
< [JeodVector](#)< ElemType >
, ElemType > [type](#)
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 45 of file object_vector.hh.

8.8.2 Member Typedef Documentation

8.8.2.1 template<typename ElemType > typedef [JeodObjectContainer](#)<[JeodVector](#)<ElemType>, ElemType>
[jeod::JeodObjectVector](#)< ElemType >::type

Template typedef for a checkpointable vector of objects.

Definition at line 50 of file object_vector.hh.

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

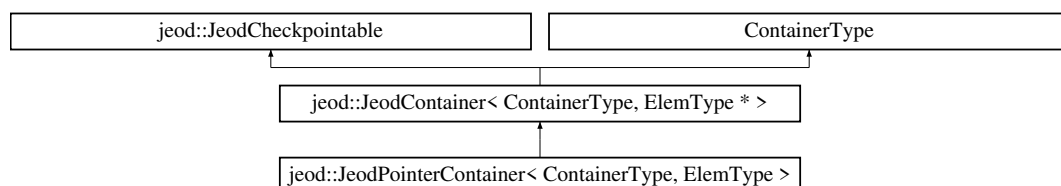
- [object_vector.hh](#)

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](#) (void)
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](#) (void)
Destruct a [JeodPointerContainer](#).
- virtual void [initialize_checkpointable](#) (const void *container, const std::type_info &container_type, const std::string elem_name)
Initialize a checkpointable object, called by the checkpoint manager.
- virtual const std::string [get_item_value](#) (void)
Return the value of the item to be written to the checkpoint file.
- virtual void [perform_insert_action](#) (const std::string &value)
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 47 of file pointer_container.hh.

8.9.2 Constructor & Destructor Documentation

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

Construct a [JeodPointerContainer](#).

Definition at line 53 of file pointer_container.hh.

8.9.2.2 `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 65 of file `pointer_container.hh`.

8.9.2.3 `template<typename ContainerType , typename ElemType > jeod::JeodPointerContainer< ContainerType, ElemType >::JeodPointerContainer (const typename ContainerType::stl_container_type & 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 77 of file `pointer_container.hh`.

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

Destruct a [JeodPointerContainer](#).

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

8.9.3 Member Function Documentation

8.9.3.1 `template<typename ContainerType , typename ElemType > virtual const std::string jeod::JeodPointerContainer< ContainerType, ElemType >::get_item_value (void)` `[inline], [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 141 of file `pointer_container.hh`.

References `jeod::JeodPointerContainer< ContainerType, ElemType >::base_type_descriptor`, and `jeod::JeodContainer< ContainerType, ElemType * >::checkpoint_iter`.

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

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

References `jeod::JeodPointerContainer< ContainerType, ElemType >::base_type_descriptor`, and `jeod::JeodContainer< ContainerType, ElemType >::initialize_checkpointable()`.

```
8.9.3.3 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 91 of file `pointer_container.hh`.

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

```
8.9.3.4 template<typename ContainerType , typename ElemType > JeodPointerContainer& jeod::JeodPointer-
        Container< 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 104 of file `pointer_container.hh`.

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

```
8.9.3.5 template<typename ContainerType , typename ElemType > virtual void jeod::JeodPointerContainer<
        ContainerType, ElemType >::perform_insert_action ( const std::string & value ) [inline],[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 154 of file `pointer_container.hh`.

8.9.4 Field Documentation

```
8.9.4.1 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 169 of file `pointer_container.hh`.

Referenced by `jeod::JeodPointerContainer< ContainerType, ElemType >::get_item_value()`, and `jeod::JeodPointerContainer< 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

- typedef [JeodPointerContainer](#)
< [JeodList](#)< ElemType * >
, ElemType > [type](#)
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 45 of file pointer_list.hh.

8.10.2 Member Typedef Documentation

8.10.2.1 `template<typename ElemType > typedef JeodPointerContainer<JeodList<ElemType*>, ElemType>
jeod::JeodPointerList< ElemType >::type`

Template typedef for a checkpointable list of pointers.

Definition at line 50 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

- typedef [JeodPointerContainer](#)
< [JeodSet](#)< ElemType * >
, ElemType > [type](#)
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 45 of file [pointer_set.hh](#).

8.11.2 Member Typedef Documentation

8.11.2.1 `template<typename ElemType > typedef JeodPointerContainer<JeodSet<ElemType*>, ElemType>
jeod::JeodPointerSet< ElemType >::type`

Template typedef for a checkpointable set of pointers.

Definition at line 50 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

- typedef [JeodPointerContainer](#)
< [JeodVector](#)< ElemType * >
, ElemType > [type](#)

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 45 of file [pointer_vector.hh](#).

8.12.2 Member Typedef Documentation

8.12.2.1 `template<typename ElemType > typedef JeodPointerContainer<JeodVector<ElemType*>, ElemType>
jeod::JeodPointerVector< ElemType >::type`

Template typedef for a checkpointable vector of pointers.

Definition at line 50 of file [pointer_vector.hh](#).

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

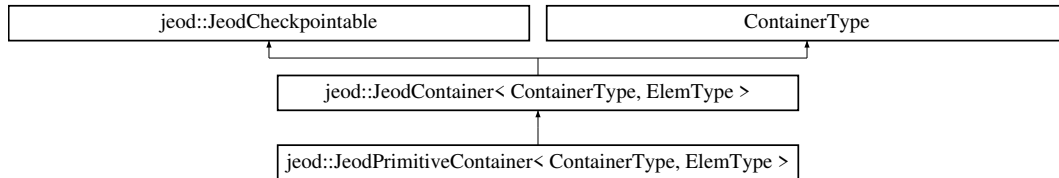
- [pointer_vector.hh](#)

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](#) (void)
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](#) (void)
Destruct a [JeodPrimitiveContainer](#).
- virtual const std::string [get_item_value](#) (void)
Return the value of the item to be written to the checkpoint file.
- virtual void [perform_insert_action](#) (const std::string &value)
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 48 of file `primitive_container.hh`.

8.13.2 Constructor & Destructor Documentation

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

Construct a [JeodPrimitiveContainer](#).

Definition at line 55 of file `primitive_container.hh`.

8.13.2.2 `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 63 of file `primitive_container.hh`.

8.13.2.3 `template<typename ContainerType , typename ElemType > jeod::JeodPrimitiveContainer< ContainerType, ElemType >::JeodPrimitiveContainer (const typename ContainerType::stl_container_type & 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 74 of file `primitive_container.hh`.

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

Destruct a [JeodPrimitiveContainer](#).

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

8.13.3 Member Function Documentation

8.13.3.1 `template<typename ContainerType , typename ElemType > virtual const std::string jeod::JeodPrimitiveContainer< ContainerType, ElemType >::get_item_value (void) [inline],[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 115 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 `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 87 of file primitive_container.hh.

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

8.13.3.3 `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 100 of file primitive_container.hh.

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

8.13.3.4 `template<typename ContainerType , typename ElemType > virtual void jeod::JeodPrimitiveContainer< ContainerType, ElemType >::perform_insert_action (const std::string & value) [inline],[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 124 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 `template<typename ContainerType , typename ElemType > JeodPrimitiveSerializer<ElemType> jeod::JeodPrimitiveContainer< ContainerType, ElemType >::serializer [protected]`

Serializer / deserializer.

trick_io(**)

Definition at line 137 of file primitive_container.hh.

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

- typedef [JeodPrimitiveContainer](#)
 < [JeodList](#)< ElemType >
 , ElemType > [type](#)

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 45 of file primitive_list.hh.

8.14.2 Member Typedef Documentation

8.14.2.1 `template<typename ElemType > typedef JeodPrimitiveContainer<JeodList<ElemType>, ElemType> jeod::JeodPrimitiveList< ElemType >::type`

Template typedef for a checkpointable list of primitives.

Definition at line 50 of file primitive_list.hh.

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

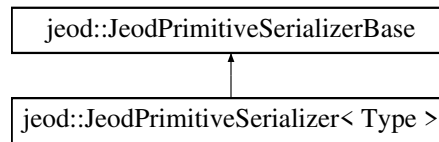
- [primitive_list.hh](#)

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](#) (void)
Construct a [JeodPrimitiveSerializer](#).
- virtual [~JeodPrimitiveSerializer](#) (void)
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.
- template<>
const std::string [to_string](#) (const std::string &val)
- template<>
std::string [from_string](#) (const std::string &val)
- template<>
const std::string [to_string](#) (const float &val)
- template<>
float [from_string](#) (const std::string &val)
- template<>
const std::string [to_string](#) (const double &val)
- template<>
double [from_string](#) (const std::string &val)
- template<>
const std::string [to_string](#) (const long double &val)
- template<>
long double [from_string](#) (const std::string &val)

Private Member Functions

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

Additional Inherited Members

8.15.1 Detailed Description

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

Serializer / deserializer for primitive data.

Definition at line 74 of file primitive_serializer.hh.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 `template<typename Type> jeod::JeodPrimitiveSerializer< Type >::JeodPrimitiveSerializer (void)`
`[inline]`

Construct a [JeodPrimitiveSerializer](#).

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

8.15.2.2 `template<typename Type> virtual jeod::JeodPrimitiveSerializer< Type >::~~JeodPrimitiveSerializer (void)`
`[inline], [virtual]`

Destruct a [JeodPrimitiveSerializer](#).

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

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

Not implemented.

8.15.3 Member Function Documentation

8.15.3.1 `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 101 of file `primitive_serializer.hh`.

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

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

Definition at line 142 of file `primitive_serializer.hh`.

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

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

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

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

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

Definition at line 214 of file `primitive_serializer.hh`.

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

Not implemented.

8.15.3.7 `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 91 of file `primitive_serializer.hh`.

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

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

Definition at line 130 of file `primitive_serializer.hh`.

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

Definition at line 154 of file `primitive_serializer.hh`.

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

Definition at line 178 of file `primitive_serializer.hh`.

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

Definition at line 202 of file `primitive_serializer.hh`.

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

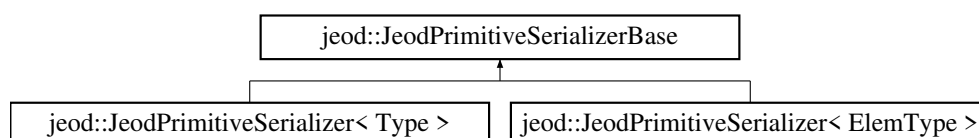
- [primitive_serializer.hh](#)

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](#) (void)
Construct a [JeodPrimitiveSerializerBase](#).
- virtual [~JeodPrimitiveSerializerBase](#) (void)
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 45 of file primitive_serializer.hh.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 `jeod::JeodPrimitiveSerializerBase::JeodPrimitiveSerializerBase (void) [inline]`

Construct a [JeodPrimitiveSerializerBase](#).

Definition at line 51 of file primitive_serializer.hh.

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

Destruct a [JeodPrimitiveSerializerBase](#).

Definition at line 56 of file primitive_serializer.hh.

8.16.3 Member Function Documentation

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

Convert a serialized double to its internal representation.

Returns

Deserialized double

Parameters

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

Definition at line 231 of file primitive_serializer.cc.

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

Convert a serialized float to its internal representation.

Returns

Deserialized float

Parameters

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

Definition at line 169 of file primitive_serializer.cc.

8.16.3.3 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

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

Definition at line 293 of file primitive_serializer.cc.

8.16.3.4 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

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

Definition at line 95 of file primitive_serializer.cc.

8.16.3.5 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 200 of file primitive_serializer.cc.

8.16.3.6 `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 138 of file primitive_serializer.cc.

8.16.3.7 `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 262 of file primitive_serializer.cc.

8.16.3.8 `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 50 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

- typedef [JeodPrimitiveContainer](#)
 < [JeodSet](#)< ElemType >
 , ElemType > [type](#)
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 45 of file primitive_set.hh.

8.17.2 Member Typedef Documentation

8.17.2.1 `template<typename ElemType > typedef JeodPrimitiveContainer<JeodSet<ElemType>, ElemType>
jeod::JeodPrimitiveSet< ElemType >::type`

Template typedef for a checkpointable set of primitives.

Definition at line 50 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

- typedef [JeodPrimitiveContainer](#)
 < [JeodVector](#)< ElemType >
 , ElemType > [type](#)

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 45 of file primitive_vector.hh.

8.18.2 Member Typedef Documentation

8.18.2.1 `template<typename ElemType > typedef JeodPrimitiveContainer<JeodVector<ElemType>, ElemType> jeod::JeodPrimitiveVector< ElemType >::type`

Template typedef for a checkpointable vector of primitives.

Definition at line 50 of file primitive_vector.hh.

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

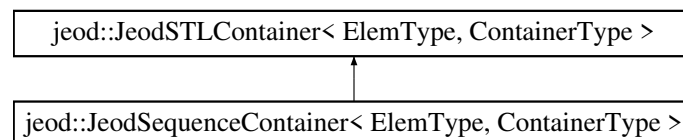
- [primitive_vector.hh](#)

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

- typedef [JeodSequenceContainer](#)
< ElemType, ContainerType > [this_container_type](#)
This type.
- typedef [JeodSTLContainer](#)
< ElemType, ContainerType > [base_container_type](#)
The JeodSTLContainer.

Public Member Functions

- virtual [~JeodSequenceContainer](#) (void)
Destructor.
- [base_container_type::reference back](#) (void)

- Get the element at the tail of the contents.*
- [base_container_type::const_reference back](#) (void) const
- Get the element at the tail of the contents.*
- [base_container_type::reference front](#) (void)
- Get the element at the head of the contents.*
- [base_container_type::const_reference front](#) (void) 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](#) (void)
- Deletes the element at the end of the contents.*

Protected Member Functions

- [JeodSequenceContainer](#) (void)
- 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 68 of file `jeod_sequence_container.hh`.

8.19.2 Member Typedef Documentation

8.19.2.1 `template<typename ElemType, typename ContainerType> typedef JeodSTLContainer<ElemType, ContainerType> jeod::JeodSequenceContainer< ElemType, ContainerType >::base_container_type`

The [JeodSTLContainer](#).

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

8.19.2.2 `template<typename ElemType, typename ContainerType> typedef JeodSequenceContainer<ElemType, ContainerType> jeod::JeodSequenceContainer< ElemType, ContainerType >::this_container_type`

This type.

Definition at line 77 of file `jeod_sequence_container.hh`.

8.19.3 Constructor & Destructor Documentation

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

Destructor.

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

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

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.

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

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

Copy constructor.

Parameters

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

Definition at line 282 of file `jeod_sequence_container.hh`.

8.19.3.4 `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 `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 149 of file jeod_sequence_container.hh.

8.19.4.2 `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 162 of file jeod_sequence_container.hh.

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

Get the element at the tail of the contents.

Definition at line 107 of file jeod_sequence_container.hh.

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

Get the element at the tail of the contents.

Definition at line 116 of file jeod_sequence_container.hh.

8.19.4.5 `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 174 of file jeod_sequence_container.hh.

8.19.4.6 `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 186 of file jeod_sequence_container.hh.

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

Get the element at the head of the contents.

Definition at line 125 of file jeod_sequence_container.hh.

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

Get the element at the head of the contents.

Definition at line 134 of file jeod_sequence_container.hh.

8.19.4.9 `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 206 of file jeod_sequence_container.hh.

8.19.4.10 `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 222 of file jeod_sequence_container.hh.

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

Deletes the element at the end of the contents.

Definition at line 258 of file jeod_sequence_container.hh.

8.19.4.12 `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 248 of file jeod_sequence_container.hh.

8.19.4.13 `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 236 of file jeod_sequence_container.hh.

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

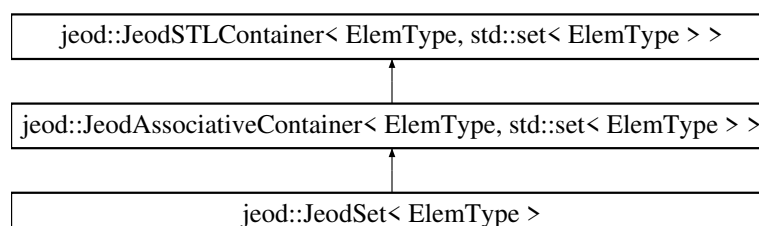
- [jeod_sequence_container.hh](#)

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

- typedef [JeodSet](#)< ElemType > [this_container_type](#)
This particular [JeodSet](#) type.
- typedef [JeodAssociativeContainer](#)
< ElemType, std::set< ElemType > > [jeod_associative_container_type](#)
The [JeodAssociativeContainer](#) type.
- typedef [JeodSTLContainer](#)
< ElemType, std::set< ElemType > > [jeod_stl_container_type](#)
The [JeodSTLContainer](#) type.
- typedef std::set< ElemType > [stl_container_type](#)
The std::set itself.

Public Member Functions

- virtual [~JeodSet](#) (void)
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](#) (void)
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 49 of file jeod_set.hh.

8.20.2 Member Typedef Documentation

8.20.2.1 template<typename ElemType > typedef [JeodAssociativeContainer](#)< ElemType, std::set<ElemType> >
[jeod::JeodSet](#)< ElemType >::[jeod_associative_container_type](#)

The [JeodAssociativeContainer](#) type.

Definition at line 65 of file jeod_set.hh.

```
8.20.2.2  template<typename ElemType > typedef JeodSTLContainer<ElemType, std::set<ElemType> >
          jeod::JeodSet< ElemType >::jeod_stl_container_type
```

The [JeodSTLContainer](#) type.

Definition at line 71 of file jeod_set.hh.

```
8.20.2.3  template<typename ElemType > typedef std::set<ElemType> jeod::JeodSet< ElemType
          >::stl_container_type
```

The std::set itself.

Definition at line 76 of file jeod_set.hh.

```
8.20.2.4  template<typename ElemType > typedef JeodSet<ElemType> jeod::JeodSet< ElemType
          >::this_container_type
```

This particular [JeodSet](#) type.

Definition at line 59 of file jeod_set.hh.

8.20.3 Constructor & Destructor Documentation

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

Destructor.

Definition at line 87 of file jeod_set.hh.

```
8.20.3.2  template<typename ElemType > jeod::JeodSet< ElemType >::JeodSet ( void ) [inline],
          [protected]
```

Default constructor.

Definition at line 117 of file jeod_set.hh.

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

Copy constructor.

Definition at line 122 of file jeod_set.hh.

```
8.20.3.4  template<typename ElemType > jeod::JeodSet< ElemType >::JeodSet ( 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 130 of file jeod_set.hh.

8.20.4 Member Function Documentation

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

Copy contents from the given source.

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

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

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

Copy contents from the given source.

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

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

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

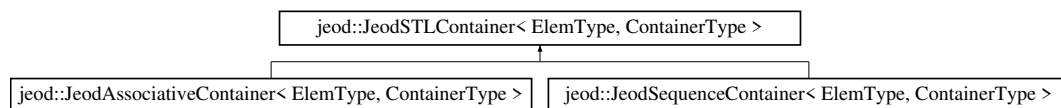
- [jeod_set.hh](#)

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

- typedef [JeodSTLContainer](#)
`< ElemType, ContainerType > this_container_type`
This particular [JeodSTLContainer](#) type.
- typedef
`ContainerType::allocator_type allocator_type`
Import the `ContainerType::allocator_type`.
- typedef `ContainerType::reference reference`
Import the `ContainerType::reference`.
- typedef
`ContainerType::const_reference const_reference`
Import the `ContainerType::const_reference`.
- typedef `ContainerType::iterator iterator`
Import the `ContainerType::iterator`.
- typedef
`ContainerType::const_iterator const_iterator`
Import the `ContainerType::const_iterator`.

- typedef
ContainerType::reverse_iterator [reverse_iterator](#)
Import the ContainerType::reverse_iterator.
- typedef
ContainerType::const_reverse_iterator [const_reverse_iterator](#)
Import the ContainerType::const_reverse_iterator.
- typedef
ContainerType::difference_type [difference_type](#)
Import the ContainerType::difference_type.
- typedef ContainerType::size_type [size_type](#)
Import the ContainerType::size_type.
- typedef ContainerType::value_type [value_type](#)
Import the ContainerType::value_type.

Public Member Functions

- virtual [~JeodSTLContainer](#) (void)
Destructor.
- [operator ContainerType &](#) (void)
Returns the contents as an lvalue.
- [operator const ContainerType &](#) (void) 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](#) (void) const
Returns the allocator object used to construct the contents.
- [iterator begin](#) (void)
Returns an iterator that points to the first element.
- [const_iterator begin](#) (void) const
Returns a const iterator that points to the first element.
- [iterator end](#) (void)
Returns an iterator that points past the last element.
- [const_iterator end](#) (void) const
Returns a const iterator that points past the last element.
- [reverse_iterator rbegin](#) (void)
Returns a reverse iterator that points to the last element.
- [const_reverse_iterator rbegin](#) (void) const
Returns a const reverse iterator that points to the last element.
- [reverse_iterator rend](#) (void)
Returns a reverse iterator that points before the first element.
- [const_reverse_iterator rend](#) (void) const
Returns a const reverse iterator that points before the first element.
- bool [empty](#) (void) const
Returns true if the contents are empty, false otherwise.
- [size_type max_size](#) (void) const
Returns the implementation's limit on the number of elements.
- [size_type size](#) (void) const
Returns the number of elements.
- void [clear](#) (void)

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` (void)
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 67 of file `jeod_stl_container.hh`.

8.21.2 Member Typedef Documentation

8.21.2.1 `template<typename ElemType, typename ContainerType> typedef ContainerType::allocator_type jeod::JeodSTLContainer< ElemType, ContainerType >::allocator_type`

Import the `ContainerType::allocator_type`.

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

```
8.21.2.2  template<typename ElemType, typename ContainerType> typedef ContainerType::const_iterator  
         jeod::JeodSTLContainer< ElemType, ContainerType >::const_iterator
```

Import the ContainerType::const_iterator.

Definition at line 103 of file jeod_stl_container.hh.

```
8.21.2.3  template<typename ElemType, typename ContainerType> typedef ContainerType::const_reference  
         jeod::JeodSTLContainer< ElemType, ContainerType >::const_reference
```

Import the ContainerType::const_reference.

Definition at line 92 of file jeod_stl_container.hh.

```
8.21.2.4  template<typename ElemType, typename ContainerType> typedef ContainerType::const_reverse_iterator  
         jeod::JeodSTLContainer< ElemType, ContainerType >::const_reverse_iterator
```

Import the ContainerType::const_reverse_iterator.

Definition at line 114 of file jeod_stl_container.hh.

```
8.21.2.5  template<typename ElemType, typename ContainerType> typedef ContainerType::difference_type  
         jeod::JeodSTLContainer< ElemType, ContainerType >::difference_type
```

Import the ContainerType::difference_type.

Definition at line 120 of file jeod_stl_container.hh.

```
8.21.2.6  template<typename ElemType, typename ContainerType> typedef ContainerType::iterator  
         jeod::JeodSTLContainer< ElemType, ContainerType >::iterator
```

Import the ContainerType::iterator.

Definition at line 98 of file jeod_stl_container.hh.

```
8.21.2.7  template<typename ElemType, typename ContainerType> typedef ContainerType::reference  
         jeod::JeodSTLContainer< ElemType, ContainerType >::reference
```

Import the ContainerType::reference.

Definition at line 87 of file jeod_stl_container.hh.

```
8.21.2.8  template<typename ElemType, typename ContainerType> typedef ContainerType::reverse_iterator  
         jeod::JeodSTLContainer< ElemType, ContainerType >::reverse_iterator
```

Import the ContainerType::reverse_iterator.

Definition at line 108 of file jeod_stl_container.hh.

```
8.21.2.9  template<typename ElemType, typename ContainerType> typedef ContainerType::size_type  
         jeod::JeodSTLContainer< ElemType, ContainerType >::size_type
```

Import the ContainerType::size_type.

Definition at line 125 of file jeod_stl_container.hh.

8.21.2.10 `template<typename ElemType, typename ContainerType> typedef JeodSTLContainer<ElemType, ContainerType> jeod::JeodSTLContainer< ElemType, ContainerType >::this_container_type`

This particular [JeodSTLContainer](#) type.

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

8.21.2.11 `template<typename ElemType, typename ContainerType> typedef ContainerType::value_type jeod::JeodSTLContainer< ElemType, ContainerType >::value_type`

Import the `ContainerType::value_type`.

Definition at line 130 of file `jeod_stl_container.hh`.

8.21.3 Constructor & Destructor Documentation

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

Destructor.

Definition at line 146 of file `jeod_stl_container.hh`.

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

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.

Definition at line 355 of file `jeod_stl_container.hh`.

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

Copy constructor.

Parameters

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

Definition at line 361 of file `jeod_stl_container.hh`.

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

Copy constructor from STL container.

Parameters

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

Definition at line 369 of file `jeod_stl_container.hh`.

8.21.4 Member Function Documentation

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

Returns an iterator that points to the first element.

Definition at line 217 of file `jeod_stl_container.hh`.

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

Returns a const iterator that points to the first element.

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

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

Clear the contents.

Definition at line 322 of file `jeod_stl_container.hh`.

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

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

Returns true if the contents are empty, false otherwise.

Definition at line 292 of file `jeod_stl_container.hh`.

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

Returns an iterator that points past the last element.

Definition at line 235 of file `jeod_stl_container.hh`.

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

Returns a const iterator that points past the last element.

Definition at line 244 of file `jeod_stl_container.hh`.

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

Returns the allocator object used to construct the contents.

Definition at line 205 of file `jeod_stl_container.hh`.

8.21.4.8 `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 335 of file jeod_stl_container.hh.

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

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

Definition at line 301 of file jeod_stl_container.hh.

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

Returns the contents as a const rvalue.

Definition at line 162 of file jeod_stl_container.hh.

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

Returns the contents as an lvalue.

Definition at line 154 of file jeod_stl_container.hh.

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

Assignment operator.

Parameters

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

Definition at line 175 of file jeod_stl_container.hh.

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

Assignment operator.

Parameters

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

Definition at line 189 of file jeod_stl_container.hh.

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

Returns a reverse iterator that points to the last element.

Definition at line 253 of file jeod_stl_container.hh.

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

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

Definition at line 262 of file jeod_stl_container.hh.

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

Returns a reverse iterator that points before the first element.

Definition at line 271 of file jeod_stl_container.hh.

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

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

Definition at line 280 of file jeod_stl_container.hh.

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

Returns the number of elements.

Definition at line 310 of file jeod_stl_container.hh.

8.21.4.19 `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 383 of file jeod_stl_container.hh.

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

The STL container.

`trick_io(**)`

Definition at line 404 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:

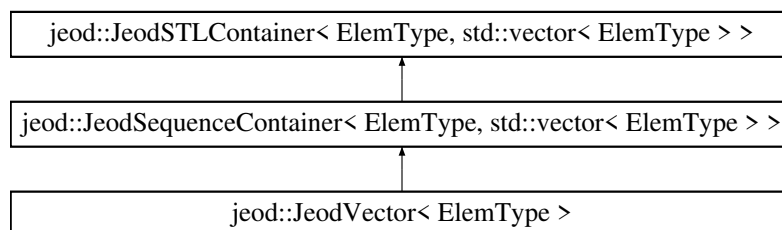
- [jeod_stl_container.hh](#)

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

- `typedef JeodVector< ElemType > this_container_type`

This particular `JeodVector` type.

- typedef [JeodSequenceContainer](#)
 < ElemType, std::vector
 < ElemType > > [jeod_sequence_container_type](#)
The [JeodSequenceContainer](#) type.
- typedef [JeodSTLContainer](#)
 < ElemType, std::vector
 < ElemType > > [jeod_stl_container_type](#)
The [JeodSTLContainer](#) type.
- typedef std::vector< ElemType > [stl_container_type](#)
The std::vector itself.

Public Member Functions

- virtual [~JeodVector](#) (void)
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](#) (void) 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](#) (void)
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 55 of file [jeod_vector.hh](#).

8.22.2 Member Typedef Documentation

8.22.2.1 `template<typename ElemType > typedef JeodSequenceContainer< ElemType, std::vector<ElemType> > jeod::JeodVector< ElemType >::jeod_sequence_container_type`

The [JeodSequenceContainer](#) type.

Definition at line 71 of file `jeod_vector.hh`.

8.22.2.2 `template<typename ElemType > typedef JeodSTLContainer<ElemType, std::vector<ElemType> > jeod::JeodVector< ElemType >::jeod_stl_container_type`

The [JeodSTLContainer](#) type.

Definition at line 77 of file `jeod_vector.hh`.

8.22.2.3 `template<typename ElemType > typedef std::vector<ElemType> jeod::JeodVector< ElemType >::stl_container_type`

The `std::vector` itself.

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

8.22.2.4 `template<typename ElemType > typedef JeodVector<ElemType> jeod::JeodVector< ElemType >::this_container_type`

This particular [JeodVector](#) type.

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

8.22.3 Constructor & Destructor Documentation

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

Destructor.

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

8.22.3.2 `template<typename ElemType > jeod::JeodVector< ElemType >::JeodVector (void) [inline], [protected]`

Default constructor.

Definition at line 189 of file `jeod_vector.hh`.

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

Copy constructor.

Definition at line 194 of file `jeod_vector.hh`.

8.22.3.4 `template<typename ElemType > jeod::JeodVector< ElemType >::JeodVector (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 202 of file jeod_vector.hh.

8.22.4 Member Function Documentation

8.22.4.1 `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 168 of file jeod_vector.hh.

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

8.22.4.2 `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 178 of file jeod_vector.hh.

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

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

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

Definition at line 124 of file jeod_vector.hh.

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

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

Copy contents from the given source.

Definition at line 101 of file jeod_vector.hh.

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

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

Copy contents from the given source.

Definition at line 111 of file jeod_vector.hh.

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

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

Get the *n*th element of the vector.

Returns

Nth element of the vector.

Definition at line 148 of file `jeod_vector.hh`.

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

8.22.4.7 `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 158 of file `jeod_vector.hh`.

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

8.22.4.8 `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 134 of file `jeod_vector.hh`.

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

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

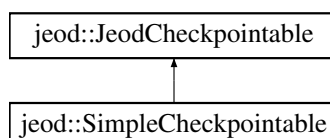
- [jeod_vector.hh](#)

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](#) ()
Construct a [SimpleCheckpointable](#) object.

- virtual `~SimpleCheckpointable` (void)
Destruct a `SimpleCheckpointable` object.
- virtual const std::string `get_init_name` (void)
Return the name of the initial restart action, in this case "restore".
- virtual const std::string `get_item_name` (void)
Return the name of the current restart action, in this case "".
- virtual const std::string `get_item_value` (void)
Return the value of the current restart action, in this case "".
- virtual void `start_checkpoint` (void)
In general, start the checkpoint process.
- virtual void `advance_checkpoint` (void)
In general, advance to the next checkpoint item; in this case, do nothing.
- virtual bool `is_checkpoint_finished` (void)
In general, indicate when checkpointing is complete.
- virtual int `perform_restore_action` (const std::string &action_name, const std::string &action_value)
In general, respond to the actions recorded in the checkpoint file.

Protected Member Functions

- virtual void `simple_restore` (void)=0
Perform the sole restore action.

Private Member Functions

- `SimpleCheckpointable` (const `SimpleCheckpointable` &)
Not implemented.
- `SimpleCheckpointable` & `operator=` (const `SimpleCheckpointable` &)
Not implemented.

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 52 of file `simple_checkpointable.hh`.

8.23.2 Constructor & Destructor Documentation

8.23.2.1 `jeod::SimpleCheckpointable::SimpleCheckpointable ()` `[inline]`

Construct a `SimpleCheckpointable` object.

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

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

Destruct a [SimpleCheckpointable](#) object.

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

8.23.2.3 `jeod::SimpleCheckpointable::SimpleCheckpointable (const SimpleCheckpointable &) [private]`

Not implemented.

8.23.3 Member Function Documentation

8.23.3.1 `virtual void jeod::SimpleCheckpointable::advance_checkpoint (void) [inline],[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 101 of file `simple_checkpointable.hh`.

8.23.3.2 `virtual const std::string jeod::SimpleCheckpointable::get_init_name (void) [inline],[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 72 of file `simple_checkpointable.hh`.

8.23.3.3 `virtual const std::string jeod::SimpleCheckpointable::get_item_name (void) [inline],[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 81 of file `simple_checkpointable.hh`.

8.23.3.4 `virtual const std::string jeod::SimpleCheckpointable::get_item_value (void) [inline],[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 88 of file `simple_checkpointable.hh`.

8.23.3.5 `virtual bool jeod::SimpleCheckpointable::is_checkpoint_finished (void) [inline],[virtual]`

In general, indicate when checkpointing is complete.

For this class, always return true.

Implements [jeod::JeodCheckpointable](#).

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

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

Not implemented.

8.23.3.7 `virtual int jeod::SimpleCheckpointable::perform_restore_action (const std::string & action_name, const std::string & action_value)` `[inline], [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 119 of file `simple_checkpointable.hh`.

References `simple_restore()`.

8.23.3.8 `virtual void jeod::SimpleCheckpointable::simple_restore (void)` `[protected], [pure virtual]`

Perform the sole restore action.

Referenced by `perform_restore_action()`.

8.23.3.9 `virtual void jeod::SimpleCheckpointable::start_checkpoint (void)` `[inline], [virtual]`

In general, start the checkpoint process.

For this class, do nothing.

Implements [jeod::JeodCheckpointable](#).

Definition at line 94 of file `simple_checkpointable.hh`.

8.23.4 Friends And Related Function Documentation

8.23.4.1 `void init_attrjeod__SimpleCheckpointable ()` `[friend]`

8.23.4.2 `friend class InputProcessor` `[friend]`

Definition at line 54 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.

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

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 <string>
#include <cstring>
#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.

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

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.

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

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::JeodSTL-Container](#)< 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::JeodSTL-Container](#)< 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::JeodSTL-Container](#)< 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::JeodSTL-Container](#)< 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::JeodSTL-Container](#)< 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.

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

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.

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

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.

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

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.

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

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.

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

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 <cstdint>
#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.

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

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 <cstdlib>
#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.

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

9.10.2 Macro Definition Documentation

- 9.10.2.1 `#define JEOD_OBJECT_CONTAINER(container_type, elem_type) JeodObjectContainer<Jeod##container_type<elem_type>,elem_type>`

Definition at line 259 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.

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

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.

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

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.

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

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.

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

9.14.2 Macro Definition Documentation

9.14.2.1 `#define JEOD_POINTER_CONTAINER(container_type, elem_type) JeodPointerContainer<Jeod##container_type<elem_type*>,elem_type>`

Definition at line 176 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.

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

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.

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

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.

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

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`.

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

9.18.2 Macro Definition Documentation

- 9.18.2.1 `#define JEOD_PRIMITIVE_CONTAINER(container_type, elem_type) JeodPrimitiveContainer<Jeod##container_type<elem_type>,elem_type>`

Definition at line 144 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.

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

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.

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

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.

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

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.

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

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.

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

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.

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

Index

- ~JeodAssociativeContainer
 - jeod::JeodAssociativeContainer, [30](#)
- ~JeodCheckpointable
 - jeod::JeodCheckpointable, [34](#)
- ~JeodContainer
 - jeod::JeodContainer, [41](#)
- ~JeodList
 - jeod::JeodList, [47](#)
- ~JeodObjectContainer
 - jeod::JeodObjectContainer, [54](#)
- ~JeodPointerContainer
 - jeod::JeodPointerContainer, [61](#)
- ~JeodPrimitiveContainer
 - jeod::JeodPrimitiveContainer, [66](#)
- ~JeodPrimitiveSerializer
 - jeod::JeodPrimitiveSerializer, [70](#)
- ~JeodPrimitiveSerializerBase
 - jeod::JeodPrimitiveSerializerBase, [72](#)
- ~JeodSTLContainer
 - jeod::JeodSTLContainer, [89](#)
- ~JeodSequenceContainer
 - jeod::JeodSequenceContainer, [78](#)
- ~JeodSet
 - jeod::JeodSet, [84](#)
- ~JeodVector
 - jeod::JeodVector, [95](#)
- ~SimpleCheckpointable
 - jeod::SimpleCheckpointable, [99](#)
- __USE_ISOC99
 - Container, [15](#)
- advance_checkpoint
 - jeod::JeodCheckpointable, [35](#)
 - jeod::JeodContainer, [41](#)
 - jeod::JeodObjectContainer, [54](#)
 - jeod::SimpleCheckpointable, [100](#)
- allocator_type
 - jeod::JeodSTLContainer, [87](#)
- assign
 - jeod::JeodSequenceContainer, [80](#)
- at
 - jeod::JeodVector, [97](#)
- back
 - jeod::JeodSequenceContainer, [80](#)
- base_container_type
 - jeod::JeodAssociativeContainer, [29](#)
 - jeod::JeodSequenceContainer, [78](#)
- base_type_descriptor
 - jeod::JeodPointerContainer, [62](#)
- begin
 - jeod::JeodSTLContainer, [89](#), [90](#)
- capacity
 - jeod::JeodVector, [97](#)
- checkpoint_iter
 - jeod::JeodContainer, [44](#)
- checkpointable.hh, [103](#)
- clear
 - jeod::JeodSTLContainer, [90](#)
- const_iterator
 - jeod::JeodSTLContainer, [87](#)
- const_reference
 - jeod::JeodSTLContainer, [88](#)
- const_reverse_iterator
 - jeod::JeodSTLContainer, [88](#)
- Container, [13](#)
 - __USE_ISOC99, [15](#)
 - operator<, [16](#)
 - operator<=, [18](#)
 - operator>, [20](#), [22](#)
 - operator>=, [22](#)
 - operator==, [18](#), [20](#)
- container.hh, [103](#)
- contents
 - jeod::JeodSTLContainer, [92](#)
- copy
 - jeod::JeodObjectContainer, [57](#)
- count
 - jeod::JeodAssociativeContainer, [30](#)
- deserialize_double
 - jeod::JeodPrimitiveSerializerBase, [72](#)
- deserialize_float
 - jeod::JeodPrimitiveSerializerBase, [73](#)
- deserialize_long_double
 - jeod::JeodPrimitiveSerializerBase, [73](#)
- deserialize_string
 - jeod::JeodPrimitiveSerializerBase, [73](#)
- difference_type
 - jeod::JeodSTLContainer, [88](#)
- elem_type_descriptor
 - jeod::JeodContainer, [44](#)
- empty
 - jeod::JeodSTLContainer, [90](#)
- end
 - jeod::JeodSTLContainer, [90](#)
- equal_range
 - jeod::JeodAssociativeContainer, [30](#), [31](#)

- erase
 - jeod::JeodAssociativeContainer, 31
 - jeod::JeodSequenceContainer, 80, 81
- find
 - jeod::JeodAssociativeContainer, 31
- from_string
 - jeod::JeodPrimitiveSerializer, 70
- front
 - jeod::JeodSequenceContainer, 81
- get_allocator
 - jeod::JeodSTLContainer, 90
- get_final_name
 - jeod::JeodCheckpointable, 35
 - jeod::JeodContainer, 41
- get_final_value
 - jeod::JeodCheckpointable, 35
 - jeod::JeodObjectContainer, 54
- get_init_name
 - jeod::JeodCheckpointable, 35
 - jeod::JeodContainer, 41
 - jeod::SimpleCheckpointable, 100
- get_init_value
 - jeod::JeodCheckpointable, 35
- get_item_name
 - jeod::JeodCheckpointable, 36
 - jeod::JeodContainer, 41
 - jeod::SimpleCheckpointable, 100
- get_item_value
 - jeod::JeodCheckpointable, 36
 - jeod::JeodObjectContainer, 54
 - jeod::JeodPointerContainer, 61
 - jeod::JeodPrimitiveContainer, 66
 - jeod::SimpleCheckpointable, 100
- index
 - jeod::JeodObjectContainer, 57
- init_attrjeod__JeodCheckpointable
 - jeod::JeodCheckpointable, 38
- init_attrjeod__JeodContainer
 - jeod::JeodContainer, 44
- init_attrjeod__JeodObjectContainer
 - jeod::JeodObjectContainer, 56
- init_attrjeod__SimpleCheckpointable
 - jeod::SimpleCheckpointable, 101
- initialize_checkpointable
 - jeod::JeodCheckpointable, 36
 - jeod::JeodContainer, 42
 - jeod::JeodPointerContainer, 61
- InputProcessor
 - jeod::JeodCheckpointable, 38
 - jeod::JeodContainer, 44
 - jeod::JeodObjectContainer, 56
 - jeod::SimpleCheckpointable, 101
- insert
 - jeod::JeodAssociativeContainer, 31, 32
 - jeod::JeodSequenceContainer, 81
 - jeod::JeodSTLContainer, 90
- is_checkpoint_finished
 - jeod::JeodCheckpointable, 36
 - jeod::JeodContainer, 42
 - jeod::SimpleCheckpointable, 100
- iterator
 - jeod::JeodSTLContainer, 88
- jeod, 25
 - jeod::JeodAssociativeContainer
 - ~JeodAssociativeContainer, 30
 - base_container_type, 29
 - count, 30
 - equal_range, 30, 31
 - erase, 31
 - find, 31
 - insert, 31, 32
 - JeodAssociativeContainer, 30
 - key_comp, 32
 - key_compare, 29
 - key_type, 29
 - lower_bound, 32
 - this_container_type, 29
 - upper_bound, 32
 - value_comp, 33
 - value_compare, 29
 - jeod::JeodAssociativeContainer< ElemType, Container-
Type >, 27
 - jeod::JeodCheckpointable, 33
 - ~JeodCheckpointable, 34
 - advance_checkpoint, 35
 - get_final_name, 35
 - get_final_value, 35
 - get_init_name, 35
 - get_init_value, 35
 - get_item_name, 36
 - get_item_value, 36
 - init_attrjeod__JeodCheckpointable, 38
 - initialize_checkpointable, 36
 - InputProcessor, 38
 - is_checkpoint_finished, 36
 - JeodCheckpointable, 34, 35
 - operator=, 36
 - perform_restore_action, 36
 - post_checkpoint, 37
 - post_restart, 37
 - pre_checkpoint, 37
 - pre_restart, 37
 - start_checkpoint, 38
 - undo_initialize_checkpointable, 38
 - jeod::JeodContainer
 - ~JeodContainer, 41
 - advance_checkpoint, 41
 - checkpoint_iter, 44
 - elem_type_descriptor, 44
 - get_final_name, 41
 - get_init_name, 41
 - get_item_name, 41
 - init_attrjeod__JeodContainer, 44
 - initialize_checkpointable, 42

- InputProcessor, 44
- is_checkpoint_finished, 42
- JeodContainer, 40
- operator=, 42
- perform_cleanup_action, 43
- perform_insert_action, 43
- perform_restore_action, 43
- start_checkpoint, 43
- stl_container_type, 40
- swap_contents, 44
- this_container_type, 40
- jeod::JeodContainer< ContainerType, ElemType >, 38
- jeod::JeodList
 - ~JeodList, 47
 - jeod_sequence_container_type, 47
 - jeod_stl_container_type, 47
 - JeodList, 47
 - merge, 49
 - operator=, 49
 - pop_front, 49
 - push_front, 49
 - remove, 50
 - remove_if, 50
 - reverse, 50
 - sort, 50
 - splice, 51
 - stl_container_type, 47
 - this_container_type, 47
 - unique, 51
- jeod::JeodList< ElemType >, 45
- jeod::JeodObjectContainer
 - ~JeodObjectContainer, 54
 - advance_checkpoint, 54
 - copy, 57
 - get_final_value, 54
 - get_item_value, 54
 - index, 57
 - init_attrjeod__JeodObjectContainer, 56
 - InputProcessor, 56
 - JeodObjectContainer, 53
 - operator=, 54, 55
 - perform_cleanup_action, 55
 - perform_insert_action, 55
 - post_checkpoint, 55
 - post_restart, 56
 - pre_checkpoint, 56
 - start_checkpoint, 56
- jeod::JeodObjectContainer< ContainerType, ElemType >, 52
- jeod::JeodObjectList
 - type, 57
- jeod::JeodObjectList< ElemType >, 57
- jeod::JeodObjectSet
 - type, 58
- jeod::JeodObjectSet< ElemType >, 58
- jeod::JeodObjectVector
 - type, 59
- jeod::JeodObjectVector< ElemType >, 58
- jeod::JeodPointerContainer
 - ~JeodPointerContainer, 61
 - base_type_descriptor, 62
 - get_item_value, 61
 - initialize_checkpointable, 61
 - JeodPointerContainer, 60, 61
 - operator=, 61, 62
 - perform_insert_action, 62
- jeod::JeodPointerContainer< ContainerType, ElemType >, 59
- jeod::JeodPointerList
 - type, 63
- jeod::JeodPointerList< ElemType >, 63
- jeod::JeodPointerSet
 - type, 64
- jeod::JeodPointerSet< ElemType >, 63
- jeod::JeodPointerVector
 - type, 64
- jeod::JeodPointerVector< ElemType >, 64
- jeod::JeodPrimitiveContainer
 - ~JeodPrimitiveContainer, 66
 - get_item_value, 66
 - JeodPrimitiveContainer, 66
 - operator=, 67
 - perform_insert_action, 67
 - serializer, 67
- jeod::JeodPrimitiveContainer< ContainerType, ElemType >, 65
- jeod::JeodPrimitiveList
 - type, 68
- jeod::JeodPrimitiveList< ElemType >, 68
- jeod::JeodPrimitiveSerializer
 - ~JeodPrimitiveSerializer, 70
 - from_string, 70
 - JeodPrimitiveSerializer, 70
 - operator=, 70
 - to_string, 71
- jeod::JeodPrimitiveSerializer< Type >, 68
- jeod::JeodPrimitiveSerializerBase, 71
 - ~JeodPrimitiveSerializerBase, 72
 - deserialize_double, 72
 - deserialize_float, 73
 - deserialize_long_double, 73
 - deserialize_string, 73
 - JeodPrimitiveSerializerBase, 72
 - serialize_double, 73
 - serialize_float, 74
 - serialize_long_double, 74
 - serialize_string, 74
- jeod::JeodPrimitiveSet
 - type, 75
- jeod::JeodPrimitiveSet< ElemType >, 75
- jeod::JeodPrimitiveVector
 - type, 76
- jeod::JeodPrimitiveVector< ElemType >, 75
- jeod::JeodSTLContainer
 - ~JeodSTLContainer, 89
 - allocator_type, 87

- begin, [89, 90](#)
- clear, [90](#)
- const_iterator, [87](#)
- const_reference, [88](#)
- const_reverse_iterator, [88](#)
- contents, [92](#)
- difference_type, [88](#)
- empty, [90](#)
- end, [90](#)
- get_allocator, [90](#)
- insert, [90](#)
- iterator, [88](#)
- JeodSTLContainer, [89](#)
- max_size, [91](#)
- operator const ContainerType &, [91](#)
- operator ContainerType &, [91](#)
- operator=, [91](#)
- rbegin, [91, 92](#)
- reference, [88](#)
- rend, [92](#)
- reverse_iterator, [88](#)
- size, [92](#)
- size_type, [88](#)
- swap, [92](#)
- this_container_type, [88](#)
- value_type, [89](#)
- jeod::JeodSTLContainer< ElemType, ContainerType >, [85](#)
- jeod::JeodSequenceContainer
 - ~JeodSequenceContainer, [78](#)
 - assign, [80](#)
 - back, [80](#)
 - base_container_type, [78](#)
 - erase, [80, 81](#)
 - front, [81](#)
 - insert, [81](#)
 - JeodSequenceContainer, [78](#)
 - pop_back, [82](#)
 - push_back, [82](#)
 - resize, [82](#)
 - this_container_type, [78](#)
- jeod::JeodSequenceContainer< ElemType, ContainerType >, [76](#)
- jeod::JeodSet
 - ~JeodSet, [84](#)
 - jeod_associative_container_type, [83](#)
 - jeod_stl_container_type, [83](#)
 - JeodSet, [84](#)
 - operator=, [85](#)
 - stl_container_type, [84](#)
 - this_container_type, [84](#)
- jeod::JeodSet< ElemType >, [82](#)
- jeod::JeodVector
 - ~JeodVector, [95](#)
 - at, [97](#)
 - capacity, [97](#)
 - jeod_sequence_container_type, [95](#)
 - jeod_stl_container_type, [95](#)
 - JeodVector, [95](#)
 - operator=, [97](#)
 - reserve, [98](#)
 - stl_container_type, [95](#)
 - this_container_type, [95](#)
- jeod::JeodVector< ElemType >, [93](#)
- jeod::SimpleCheckpointable, [98](#)
 - ~SimpleCheckpointable, [99](#)
 - advance_checkpoint, [100](#)
 - get_init_name, [100](#)
 - get_item_name, [100](#)
 - get_item_value, [100](#)
 - init_attrjeod__SimpleCheckpointable, [101](#)
 - InputProcessor, [101](#)
 - is_checkpoint_finished, [100](#)
 - operator=, [100](#)
 - perform_restore_action, [101](#)
 - simple_restore, [101](#)
 - SimpleCheckpointable, [99, 100](#)
 - start_checkpoint, [101](#)
- jeod_associative_container.hh, [104](#)
- jeod_associative_container_type
 - jeod::JeodSet, [83](#)
- jeod_container_compare.hh, [104](#)
- jeod_list.hh, [106](#)
- jeod_sequence_container.hh, [107](#)
- jeod_sequence_container_type
 - jeod::JeodList, [47](#)
 - jeod::JeodVector, [95](#)
- jeod_set.hh, [107](#)
- jeod_stl_container.hh, [108](#)
- jeod_stl_container_type
 - jeod::JeodList, [47](#)
 - jeod::JeodSet, [83](#)
 - jeod::JeodVector, [95](#)
- jeod_vector.hh, [108](#)
- JeodAssociativeContainer
 - jeod::JeodAssociativeContainer, [30](#)
- JeodCheckpointable
 - jeod::JeodCheckpointable, [34, 35](#)
- JeodContainer
 - jeod::JeodContainer, [40](#)
- JeodList
 - jeod::JeodList, [47](#)
- JeodObjectContainer
 - jeod::JeodObjectContainer, [53](#)
- JeodPointerContainer
 - jeod::JeodPointerContainer, [60, 61](#)
- JeodPrimitiveContainer
 - jeod::JeodPrimitiveContainer, [66](#)
- JeodPrimitiveSerializer
 - jeod::JeodPrimitiveSerializer, [70](#)
- JeodPrimitiveSerializerBase
 - jeod::JeodPrimitiveSerializerBase, [72](#)
- JeodSTLContainer
 - jeod::JeodSTLContainer, [89](#)
- JeodSequenceContainer
 - jeod::JeodSequenceContainer, [78](#)

- JeodSet
 - jeod::JeodSet, [84](#)
- JeodVector
 - jeod::JeodVector, [95](#)
- key_comp
 - jeod::JeodAssociativeContainer, [32](#)
- key_compare
 - jeod::JeodAssociativeContainer, [29](#)
- key_type
 - jeod::JeodAssociativeContainer, [29](#)
- lower_bound
 - jeod::JeodAssociativeContainer, [32](#)
- max_size
 - jeod::JeodSTLContainer, [91](#)
- merge
 - jeod::JeodList, [49](#)
- Models, [11](#)
- object_container.hh, [109](#)
- object_list.hh, [109](#)
- object_set.hh, [110](#)
- object_vector.hh, [110](#)
- operator const ContainerType &
 - jeod::JeodSTLContainer, [91](#)
- operator ContainerType &
 - jeod::JeodSTLContainer, [91](#)
- operator<
 - Container, [16](#)
- operator<=
 - Container, [18](#)
- operator>
 - Container, [20](#), [22](#)
- operator>=
 - Container, [22](#)
- operator=
 - jeod::JeodCheckpointable, [36](#)
 - jeod::JeodContainer, [42](#)
 - jeod::JeodList, [49](#)
 - jeod::JeodObjectContainer, [54](#), [55](#)
 - jeod::JeodPointerContainer, [61](#), [62](#)
 - jeod::JeodPrimitiveContainer, [67](#)
 - jeod::JeodPrimitiveSerializer, [70](#)
 - jeod::JeodSet, [85](#)
 - jeod::JeodSTLContainer, [91](#)
 - jeod::JeodVector, [97](#)
 - jeod::SimpleCheckpointable, [100](#)
- operator==
 - Container, [18](#), [20](#)
- perform_cleanup_action
 - jeod::JeodContainer, [43](#)
 - jeod::JeodObjectContainer, [55](#)
- perform_insert_action
 - jeod::JeodContainer, [43](#)
 - jeod::JeodObjectContainer, [55](#)
 - jeod::JeodPointerContainer, [62](#)
 - jeod::JeodPrimitiveContainer, [67](#)
- perform_restore_action
 - jeod::JeodCheckpointable, [36](#)
 - jeod::JeodContainer, [43](#)
 - jeod::SimpleCheckpointable, [101](#)
- pointer_container.hh, [111](#)
- pointer_list.hh, [112](#)
- pointer_set.hh, [112](#)
- pointer_vector.hh, [113](#)
- pop_back
 - jeod::JeodSequenceContainer, [82](#)
- pop_front
 - jeod::JeodList, [49](#)
- post_checkpoint
 - jeod::JeodCheckpointable, [37](#)
 - jeod::JeodObjectContainer, [55](#)
- post_restart
 - jeod::JeodCheckpointable, [37](#)
 - jeod::JeodObjectContainer, [56](#)
- pre_checkpoint
 - jeod::JeodCheckpointable, [37](#)
 - jeod::JeodObjectContainer, [56](#)
- pre_restart
 - jeod::JeodCheckpointable, [37](#)
- primitive_container.hh, [113](#)
- primitive_list.hh, [114](#)
- primitive_serializer.cc, [114](#)
- primitive_serializer.hh, [115](#)
- primitive_set.hh, [115](#)
- primitive_vector.hh, [116](#)
- push_back
 - jeod::JeodSequenceContainer, [82](#)
- push_front
 - jeod::JeodList, [49](#)
- rbegin
 - jeod::JeodSTLContainer, [91](#), [92](#)
- reference
 - jeod::JeodSTLContainer, [88](#)
- remove
 - jeod::JeodList, [50](#)
- remove_if
 - jeod::JeodList, [50](#)
- rend
 - jeod::JeodSTLContainer, [92](#)
- reserve
 - jeod::JeodVector, [98](#)
- resize
 - jeod::JeodSequenceContainer, [82](#)
- reverse
 - jeod::JeodList, [50](#)
- reverse_iterator
 - jeod::JeodSTLContainer, [88](#)
- serialize_double
 - jeod::JeodPrimitiveSerializerBase, [73](#)
- serialize_float
 - jeod::JeodPrimitiveSerializerBase, [74](#)
- serialize_long_double

- jeod::JeodPrimitiveSerializerBase, 74
- serialize_string
 - jeod::JeodPrimitiveSerializerBase, 74
- serializer
 - jeod::JeodPrimitiveContainer, 67
- simple_checkpointable.hh, 116
- simple_restore
 - jeod::SimpleCheckpointable, 101
- SimpleCheckpointable
 - jeod::SimpleCheckpointable, 99, 100
- size
 - jeod::JeodSTLContainer, 92
- size_type
 - jeod::JeodSTLContainer, 88
- sort
 - jeod::JeodList, 50
- splice
 - jeod::JeodList, 51
- start_checkpoint
 - jeod::JeodCheckpointable, 38
 - jeod::JeodContainer, 43
 - jeod::JeodObjectContainer, 56
 - jeod::SimpleCheckpointable, 101
- stl_container_type
 - jeod::JeodContainer, 40
 - jeod::JeodList, 47
 - jeod::JeodSet, 84
 - jeod::JeodVector, 95
- swap
 - jeod::JeodSTLContainer, 92
- swap_contents
 - jeod::JeodContainer, 44
- this_container_type
 - jeod::JeodAssociativeContainer, 29
 - jeod::JeodContainer, 40
 - jeod::JeodList, 47
 - jeod::JeodSequenceContainer, 78
 - jeod::JeodSet, 84
 - jeod::JeodSTLContainer, 88
 - jeod::JeodVector, 95
- to_string
 - jeod::JeodPrimitiveSerializer, 71
- type
 - jeod::JeodObjectList, 57
 - jeod::JeodObjectSet, 58
 - jeod::JeodObjectVector, 59
 - jeod::JeodPointerList, 63
 - jeod::JeodPointerSet, 64
 - jeod::JeodPointerVector, 64
 - jeod::JeodPrimitiveList, 68
 - jeod::JeodPrimitiveSet, 75
 - jeod::JeodPrimitiveVector, 76
- undo_initialize_checkpointable
 - jeod::JeodCheckpointable, 38
- unique
 - jeod::JeodList, 51
- upper_bound
 - jeod::JeodAssociativeContainer, 32
- Utils, 12
- value_comp
 - jeod::JeodAssociativeContainer, 33
- value_compare
 - jeod::JeodAssociativeContainer, 29
- value_type
 - jeod::JeodSTLContainer, 89