

DynamicsManagerModel

5.0

Generated by Doxygen 1.8.14

Contents

1	Module Index	1
1.1	Modules	1
2	Namespace Index	3
2.1	Namespace List	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Data Structure Index	7
4.1	Data Structures	7
5	File Index	9
5.1	File List	9
6	Module Documentation	11
6.1	Models	11
6.1.1	Detailed Description	11
6.2	Dynamics	12
6.2.1	Detailed Description	12
6.3	DynManager	13
6.3.1	Detailed Description	13
7	Namespace Documentation	15
7.1	er7_utils Namespace Reference	15
7.1.1	Detailed Description	15
7.2	jeod Namespace Reference	15
7.2.1	Detailed Description	15

8 Data Structure Documentation	17
8.1 jeod::BaseDynManager Class Reference	17
8.1.1 Detailed Description	18
8.1.2 Constructor & Destructor Documentation	18
8.1.2.1 ~BaseDynManager()	18
8.1.3 Member Function Documentation	18
8.1.3.1 add_dyn_body()	18
8.1.3.2 add_integ_group()	19
8.1.3.3 add_mass_body() [1/2]	19
8.1.3.4 add_mass_body() [2/2]	19
8.1.3.5 find_dyn_body()	20
8.1.3.6 find_mass_body()	20
8.1.3.7 get_dyn_bodies()	20
8.1.3.8 initialize_gravity_controls()	21
8.1.3.9 is_dyn_body_registered()	21
8.1.3.10 is_integ_group_registered()	21
8.1.3.11 is_mass_body_registered()	22
8.1.3.12 reset_gravity_controls()	22
8.1.3.13 reset_integrators() [1/2]	22
8.1.3.14 reset_integrators() [2/2]	22
8.1.3.15 set_gravity_manager()	23
8.1.3.16 timestamp()	23
8.1.4 Friends And Related Function Documentation	23
8.1.4.1 init_attrjeod__BaseDynManager	23
8.1.4.2 InputProcessor	24
8.2 jeod::DynamicsIntegrationGroup Class Reference	24
8.2.1 Detailed Description	25
8.2.2 Constructor & Destructor Documentation	25
8.2.2.1 DynamicsIntegrationGroup() [1/3]	26
8.2.2.2 DynamicsIntegrationGroup() [2/3]	26

8.2.2.3	~DynamicsIntegrationGroup()	26
8.2.2.4	DynamicsIntegrationGroup() [3/3]	26
8.2.3	Member Function Documentation	27
8.2.3.1	add_dyn_body()	27
8.2.3.2	collect_derivatives()	27
8.2.3.3	create_group()	27
8.2.3.4	delete_dyn_body()	28
8.2.3.5	gravitation()	28
8.2.3.6	initialize_group()	29
8.2.3.7	integrate_bodies()	29
8.2.3.8	is_empty()	29
8.2.3.9	operator=()	30
8.2.3.10	prepare_for_integ_loop()	30
8.2.3.11	register_base_contents()	30
8.2.3.12	register_group()	31
8.2.3.13	reset_body_integrators()	31
8.2.4	Friends And Related Function Documentation	31
8.2.4.1	init_attrjeod__DynamicsIntegrationGroup	31
8.2.4.2	InputProcessor	31
8.2.5	Field Documentation	32
8.2.5.1	bodies_integrated_separately	32
8.2.5.2	deriv_ephem_update	32
8.2.5.3	dyn_bodies	32
8.3	jeod::DynManager Class Reference	33
8.3.1	Detailed Description	36
8.3.2	Constructor & Destructor Documentation	36
8.3.2.1	DynManager() [1/2]	36
8.3.2.2	~DynManager()	36
8.3.2.3	DynManager() [2/2]	37
8.3.3	Member Function Documentation	37

8.3.3.1	<code>add_body_action()</code>	37
8.3.3.2	<code>add_dyn_body()</code>	37
8.3.3.3	<code>add_integ_group()</code>	38
8.3.3.4	<code>add_mass_body()</code> [1/2]	38
8.3.3.5	<code>add_mass_body()</code> [2/2]	38
8.3.3.6	<code>check_for_uninitialized_states()</code>	39
8.3.3.7	<code>compute_derivatives()</code>	39
8.3.3.8	<code>find_dyn_body()</code>	39
8.3.3.9	<code>find_mass_body()</code>	40
8.3.3.10	<code>get_dyn_bodies()</code>	40
8.3.3.11	<code>gravitation()</code>	41
8.3.3.12	<code>initialize_dyn_bodies()</code>	41
8.3.3.13	<code>initialize_dyn_body()</code>	41
8.3.3.14	<code>initialize_gravity_controls()</code>	42
8.3.3.15	<code>initialize_integ_groups()</code>	42
8.3.3.16	<code>initialize_model()</code> [1/2]	42
8.3.3.17	<code>initialize_model()</code> [2/2]	43
8.3.3.18	<code>initialize_model_internal()</code>	43
8.3.3.19	<code>initialize_simulation()</code>	44
8.3.3.20	<code>integrate()</code>	44
8.3.3.21	<code>is_dyn_body_registered()</code>	44
8.3.3.22	<code>is_initialized()</code>	45
8.3.3.23	<code>is_integ_group_registered()</code>	45
8.3.3.24	<code>is_mass_body_registered()</code>	46
8.3.3.25	<code>name()</code>	46
8.3.3.26	<code>operator=()</code>	47
8.3.3.27	<code>perform_actions()</code>	47
8.3.3.28	<code>perform_dyn_body_initializations()</code>	47
8.3.3.29	<code>perform_mass_attach_initializations()</code>	47
8.3.3.30	<code>perform_mass_body_initializations()</code>	48

8.3.3.31	remove_body_action()	48
8.3.3.32	reset_gravity_controls()	48
8.3.3.33	reset_integrators() [1/2]	49
8.3.3.34	reset_integrators() [2/2]	49
8.3.3.35	set_gravity_manager()	49
8.3.3.36	shutdown()	50
8.3.3.37	timestamp()	50
8.3.3.38	update_integration_group()	50
8.3.4	Friends And Related Function Documentation	50
8.3.4.1	init_attrjeod__DynManager	51
8.3.4.2	InputProcessor	51
8.3.5	Field Documentation	51
8.3.5.1	body_actions	51
8.3.5.2	default_integ_group	51
8.3.5.3	deriv_ephem_update	52
8.3.5.4	dyn_bodies	52
8.3.5.5	gravity_manager	52
8.3.5.6	gravity_off	52
8.3.5.7	initialized	53
8.3.5.8	integ_constructor	53
8.3.5.9	integ_groups	53
8.3.5.10	integ_interface	53
8.3.5.11	mass_bodies	54
8.3.5.12	mode	54
8.3.5.13	sim_integrator	54
8.3.5.14	simple_ephemeris	54
8.4	jeod::DynManagerInit Class Reference	55
8.4.1	Detailed Description	55
8.4.2	Member Enumeration Documentation	55
8.4.2.1	EphemerisMode	55

8.4.3	Constructor & Destructor Documentation	56
8.4.3.1	DynManagerInit() [1/2]	56
8.4.3.2	~DynManagerInit()	56
8.4.3.3	DynManagerInit() [2/2]	56
8.4.4	Member Function Documentation	56
8.4.4.1	operator=()	56
8.4.5	Field Documentation	57
8.4.5.1	central_point_name	57
8.4.5.2	integ_constructor	57
8.4.5.3	integ_group_constructor	58
8.4.5.4	jeod_integ_opt	58
8.4.5.5	mode	58
8.4.5.6	sim_integ_opt	59
8.5	jeod::DynManagerMessages Class Reference	59
8.5.1	Detailed Description	60
8.5.2	Constructor & Destructor Documentation	60
8.5.2.1	DynManagerMessages() [1/2]	60
8.5.2.2	DynManagerMessages() [2/2]	60
8.5.3	Member Function Documentation	60
8.5.3.1	operator=()	60
8.5.4	Friends And Related Function Documentation	60
8.5.4.1	init_attrjeod__DynManagerMessages	61
8.5.4.2	InputProcessor	61
8.5.5	Field Documentation	61
8.5.5.1	duplicate_entry	61
8.5.5.2	inconsistent_setup	61
8.5.5.3	internal_error	62
8.5.5.4	invalid_frame	62
8.5.5.5	invalid_name	62
8.5.5.6	invalid_type	62
8.5.5.7	null_pointer	63
8.5.5.8	singleton_error	63

9 File Documentation	65
9.1 base_dyn_manager.hh File Reference	65
9.1.1 Detailed Description	65
9.2 class_declarations.hh File Reference	65
9.2.1 Detailed Description	66
9.3 dyn_bodies_primitives.cc File Reference	66
9.3.1 Detailed Description	66
9.4 dyn_manager.cc File Reference	66
9.4.1 Detailed Description	67
9.5 dyn_manager.hh File Reference	67
9.5.1 Detailed Description	67
9.6 dyn_manager_init.cc File Reference	67
9.6.1 Detailed Description	68
9.7 dyn_manager_init.hh File Reference	68
9.7.1 Detailed Description	68
9.8 dyn_manager_messages.cc File Reference	68
9.8.1 Detailed Description	69
9.8.2 Macro Definition Documentation	69
9.8.2.1 MAKE_DYNMANAGER_MESSAGE_CODE	69
9.9 dyn_manager_messages.hh File Reference	69
9.9.1 Detailed Description	69
9.10 dynamics_integration_group.cc File Reference	70
9.10.1 Detailed Description	70
9.11 dynamics_integration_group.hh File Reference	70
9.11.1 Detailed Description	71
9.12 gravitation.cc File Reference	71
9.12.1 Detailed Description	71
9.13 initialize_dyn_bodies.cc File Reference	71
9.13.1 Detailed Description	72
9.14 initialize_model.cc File Reference	72
9.14.1 Detailed Description	72
9.15 initialize_simulation.cc File Reference	72
9.15.1 Detailed Description	73
9.16 integ_group_primitives.cc File Reference	73
9.16.1 Detailed Description	73
9.17 mass_bodies_primitives.cc File Reference	73
9.17.1 Detailed Description	74
9.18 perform_actions.cc File Reference	74
9.18.1 Detailed Description	74
Index	75

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Models	11
Dynamics	12
DynManager	13

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

er7_utils	Namespace er7_utils contains the state integration models used by JEOD	15
jeod	Namespace jeod	15

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

BaseEphemeridesManager	
jeod::BaseDynManager	17
jeod::DynManager	33
jeod::DynManagerInit	55
jeod::DynManagerMessages	59
EphemeridesManager	
jeod::DynManager	33
JeodIntegrationGroup	
jeod::DynamicsIntegrationGroup	24
JeodIntegrationGroupOwner	
jeod::DynManager	33

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

jeod::BaseDynManager	
The DynManager class augments the EphemManager with dynamics-related items	17
jeod::DynamicsIntegrationGroup	
A DynamicsIntegrationGroup integrates the state of a set of DynBoby objects over time	24
jeod::DynManager	
Manages the dynamic elements of a simulation	33
jeod::DynManagerInit	
This class contains data used to initialize a DynManager object	55
jeod::DynManagerMessages	
Specifies the message IDs used in the DynManager model	59

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

base_dyn_manager.hh	Define the BaseDynManager class, which defines the interfaces to the class DynManager . . .	65
class_declarations.hh	Forward declarations of classes defined in dyn_manager.hh . . .	65
dyn_bodies_primitives.cc	Define the DynManager member functions that search through and add elements to the collection of DynBody pointers . . .	66
dyn_manager.cc	Define simple member functions for the DynManager and related classes . . .	66
dyn_manager.hh	Define the DynManager class, which manages the planets and vehicles modeled in a JEOD-based simulation . . .	67
dyn_manager_init.cc	Define member functions for the DynManagerInit class . . .	67
dyn_manager_init.hh	Define the DynManagerInit class, which contains the data used to initialize a DynManager object . . .	68
dyn_manager_messages.cc	Implement the class DynManagerMessages . . .	68
dyn_manager_messages.hh	Define the class DynManagerMessages, the class that specifies the message IDs used in the DynManager model . . .	69
dynamics_integration_group.cc	Define DynamicsIntegrationGroup methods . . .	70
dynamics_integration_group.hh	Define the extensible class DynamicsIntegrationGroup, an instance of which is responsible for integrating the states of a set of DynBody objects . . .	70
gravitation.cc	Compute gravitational acceleration . . .	71
initialize_dyn_bodies.cc	Define DynManager::initialize_dyn_bodies . . .	71
initialize_model.cc	Define DynManager::initialize_model . . .	72
initialize_simulation.cc	Define DynManager::initialize_simulation, which completes the initialization of the JEOD dynamics manager . . .	72

integ_group_primitives.cc	
Define the DynManager member functions that search through and add elements to the collection of DynamicsIntegrationGroup pointers	73
mass_bodies_primitives.cc	
Define the DynManager member functions that search through and add elements to the collection of MassBody pointers	73
perform_actions.cc	
Define DynManager::perform_actions	74

Chapter 6

Module Documentation

6.1 Models

Modules

- [Dynamics](#)

6.1.1 Detailed Description

6.2 Dynamics

Modules

- [DynManager](#)

6.2.1 Detailed Description

6.3 DynManager

Files

- file [base_dyn_manager.hh](#)
Define the BaseDynManager class, which defines the interfaces to the class DynManager.
- file [class_declarations.hh](#)
Forward declarations of classes defined in [dyn_manager.hh](#).
- file [dyn_manager.hh](#)
Define the DynManager class, which manages the planets and vehicles modeled in a JEOD-based simulation.
- file [dyn_manager_init.hh](#)
Define the DynManagerInit class, which contains the data used to initialize a DynManager object.
- file [dyn_manager_messages.hh](#)
Define the class DynManagerMessages, the class that specifies the message IDs used in the DynManager model.
- file [dynamics_integration_group.hh](#)
Define the extensible class DynamicsIntegrationGroup, an instance of which is responsible for integrating the states of a set of DynBody objects.
- file [dyn_bodies_primitives.cc](#)
Define the DynManager member functions that search through and add elements to the collection of DynBody pointers.
- file [dyn_manager.cc](#)
Define simple member functions for the DynManager and related classes.
- file [dyn_manager_init.cc](#)
Define member functions for the DynManagerInit class.
- file [dyn_manager_messages.cc](#)
Implement the class DynManagerMessages.
- file [dynamics_integration_group.cc](#)
Define DynamicsIntegrationGroup methods.
- file [gravitation.cc](#)
Compute gravitational acceleration.
- file [initialize_dyn_bodies.cc](#)
Define DynManager::initialize_dyn_bodies.
- file [initialize_model.cc](#)
Define DynManager::initialize_model.
- file [initialize_simulation.cc](#)
Define DynManager::initialize_simulation, which completes the initialization of the JEOD dynamics manager.
- file [integ_group_primitives.cc](#)
Define the DynManager member functions that search through and add elements to the collection of DynamicsIntegrationGroup pointers.
- file [mass_bodies_primitives.cc](#)
Define the DynManager member functions that search through and add elements to the collection of MassBody pointers.
- file [perform_actions.cc](#)
Define DynManager::perform_actions.

Namespaces

- [jeod](#)
Namespace jeod.
- [er7_utils](#)
Namespace [er7_utils](#) contains the state integration models used by JEOD.

6.3.1 Detailed Description

Chapter 7

Namespace Documentation

7.1 er7_utils Namespace Reference

Namespace [er7_utils](#) contains the state integration models used by JEOD.

7.1.1 Detailed Description

Namespace [er7_utils](#) contains the state integration models used by JEOD.

7.2 jeod Namespace Reference

Namespace jeod.

Data Structures

- class [BaseDynManager](#)
The [DynManager](#) class augments the [EphemManager](#) with dynamics-related items.
- class [DynamicsIntegrationGroup](#)
A [DynamicsIntegrationGroup](#) integrates the state of a set of [DynBoby](#) objects over time.
- class [DynManager](#)
The [DynManager](#) class manages the dynamic elements of a simulation.
- class [DynManagerInit](#)
This class contains data used to initialize a [DynManager](#) object.
- class [DynManagerMessages](#)
Specifies the message IDs used in the [DynManager](#) model.

7.2.1 Detailed Description

Namespace jeod.

Chapter 8

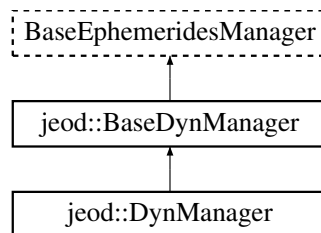
Data Structure Documentation

8.1 jeod::BaseDynManager Class Reference

The [DynManager](#) class augments the EphemManager with dynamics-related items.

```
#include <base_dyn_manager.hh>
```

Inheritance diagram for jeod::BaseDynManager:



Public Member Functions

- virtual [~BaseDynManager](#) ()
Destructor.
- virtual void [set_gravity_manager](#) (GravityManager &gravity)=0
Set the Gravity Manager.
- virtual void [initialize_gravity_controls](#) ()=0
Initialize the gravity model controls.
- virtual void [reset_gravity_controls](#) (void)=0
Reset the gravity model controls.
- virtual void [add_mass_body](#) (MassBody &mass_body)=0
Add a mass body to the list of such.
- virtual void [add_mass_body](#) (MassBody *mass_body)=0
Add a mass body to the list of such.
- virtual MassBody * [find_mass_body](#) (const char *name) const =0
Find a mass body.
- virtual bool [is_mass_body_registered](#) (const MassBody *mass_body) const =0
Check if a mass body has been registered with the dynamics manager.

- virtual void [add_dyn_body](#) (DynBody &dyn_body)=0
Add a dynamic body to the list of such.
- virtual DynBody * [find_dyn_body](#) (const char *name) const =0
Find a dynamic body.
- virtual std::vector< DynBody * > [get_dyn_bodies](#) () const =0
Return a copy of the list of registered dynamic bodies.
- virtual bool [is_dyn_body_registered](#) (const DynBody *dyn_body) const =0
Check if a dynamic body has been registered with the dynamics manager.
- virtual void [add_integ_group](#) (DynamicsIntegrationGroup &integ_group)=0
Add an integration group to the list of such.
- virtual bool [is_integ_group_registered](#) (const DynamicsIntegrationGroup *integ_group) const =0
Check if an integration group has been registered.
- virtual void [reset_integrators](#) ()=0
Force all integrators to reset themselves.
- virtual void [reset_integrators](#) (DynamicsIntegrationGroup &integ_group)=0
Instruct specific integration group to reset its integrators.
- virtual double [timestamp](#) (void) const =0
Get the time at which the manager was last updated.

Friends

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

8.1.1 Detailed Description

The [DynManager](#) class augments the EphemManager with dynamics-related items.

This class defines the external interfaces to that class.

Definition at line 84 of file base_dyn_manager.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 ~BaseDynManager()

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

Destructor.

Definition at line 100 of file base_dyn_manager.hh.

8.1.3 Member Function Documentation

8.1.3.1 add_dyn_body()

```
virtual void jeod::BaseDynManager::add_dyn_body (
    DynBody & dyn_body ) [pure virtual]
```

Add a dynamic body to the list of such.

Parameters

<i>dyn_body</i>	Body to be added to the list of dynamic bodies.
-----------------	---

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

8.1.3.2 add_integ_group()

```
virtual void jeod::BaseDynManager::add_integ_group (
    DynamicsIntegrationGroup & integ_group ) [pure virtual]
```

Add an integration group to the list of such.

Parameters

<i>integ_group</i>	Group to be added to the list of integration groups.
--------------------	--

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

8.1.3.3 add_mass_body() [1/2]

```
virtual void jeod::BaseDynManager::add_mass_body (
    MassBody & mass_body ) [pure virtual]
```

Add a mass body to the list of such.

Parameters

<i>mass_body</i>	Body to be added to the list of mass bodies.
------------------	--

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

8.1.3.4 add_mass_body() [2/2]

```
virtual void jeod::BaseDynManager::add_mass_body (
    MassBody * mass_body ) [pure virtual]
```

Add a mass body to the list of such.

Parameters

<i>mass_body</i>	Body to be added to the list of mass bodies.
------------------	--

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

8.1.3.5 find_dyn_body()

```
virtual DynBody* jeod::BaseDynManager::find_dyn_body (
    const char * name ) const [pure virtual]
```

Find a dynamic body.

Parameters

<i>name</i>	Dynamic body name.
-------------	--------------------

Returns

Pointer to the dynamic body with the given name.

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

8.1.3.6 find_mass_body()

```
virtual MassBody* jeod::BaseDynManager::find_mass_body (
    const char * name ) const [pure virtual]
```

Find a mass body.

Parameters

<i>name</i>	Mass body name.
-------------	-----------------

Returns

Pointer to the mass body with the given name.

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

8.1.3.7 get_dyn_bodies()

```
virtual std::vector<DynBody*> jeod::BaseDynManager::get_dyn_bodies ( ) const [pure virtual]
```

Return a copy of the list of registered dynamic bodies.

Returns

Copy of dyn_bodies data member

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

8.1.3.8 initialize_gravity_controls()

```
virtual void jeod::BaseDynManager::initialize_gravity_controls ( ) [pure virtual]
```

Initialize the gravity model controls.

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

8.1.3.9 is_dyn_body_registered()

```
virtual bool jeod::BaseDynManager::is_dyn_body_registered (
    const DynBody * dyn_body ) const [pure virtual]
```

Check if a dynamic body has been registered with the dynamics manager.

Parameters

<i>dyn_body</i>	Dynamic body to be checked.
-----------------	-----------------------------

Returns

True if the body is registered, false otherwise.

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

8.1.3.10 is_integ_group_registered()

```
virtual bool jeod::BaseDynManager::is_integ_group_registered (
    const DynamicsIntegrationGroup * integ_group ) const [pure virtual]
```

Check if an integration group has been registered.

Parameters

<i>integ_group</i>	Integration group to be checked.
--------------------	----------------------------------

Returns

True if the group is registered, false otherwise.

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

8.1.3.11 is_mass_body_registered()

```
virtual bool jeod::BaseDynManager::is_mass_body_registered (
    const MassBody * mass_body ) const [pure virtual]
```

Check if a mass body has been registered with the dynamics manager.

Parameters

<i>mass_body</i>	Mass body to be checked.
------------------	--------------------------

Returns

True if the body is registered, false otherwise.

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

8.1.3.12 reset_gravity_controls()

```
virtual void jeod::BaseDynManager::reset_gravity_controls (
    void ) [pure virtual]
```

Reset the gravity model controls.

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

8.1.3.13 reset_integrators() [1/2]

```
virtual void jeod::BaseDynManager::reset_integrators ( ) [pure virtual]
```

Force all integrators to reset themselves.

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

8.1.3.14 reset_integrators() [2/2]

```
virtual void jeod::BaseDynManager::reset_integrators (
    DynamicsIntegrationGroup & integ_group ) [pure virtual]
```

Instruct specific integration group to reset its integrators.

Parameters

<i>integ_group</i>	Integration group to be reset.
--------------------	--------------------------------

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

8.1.3.15 set_gravity_manager()

```
virtual void jeod::BaseDynManager::set_gravity_manager (
    GravityManager & gravity ) [pure virtual]
```

Set the Gravity Manager.

Parameters

<i>gravity</i>	link to the manager of gravity model.
----------------	---------------------------------------

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

8.1.3.16 timestamp()

```
virtual double jeod::BaseDynManager::timestamp (
    void ) const [pure virtual]
```

Get the time at which the manager was last updated.

Returns

Time at which the manager was last updated.

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

8.1.4 Friends And Related Function Documentation

8.1.4.1 init_attrjeod__BaseDynManager

```
void init_attrjeod__BaseDynManager ( ) [friend]
```

8.1.4.2 InputProcessor

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

Definition at line 87 of file base_dyn_manager.hh.

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

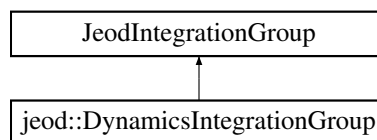
- [base_dyn_manager.hh](#)

8.2 jeod::DynamicsIntegrationGroup Class Reference

A [DynamicsIntegrationGroup](#) integrates the state of a set of DynBody objects over time.

```
#include <dynamics_integration_group.hh>
```

Inheritance diagram for jeod::DynamicsIntegrationGroup:



Public Member Functions

- [DynamicsIntegrationGroup](#) ()
DynamicsIntegrationGroup default constructor, needed for checkpoint/restart.
- [DynamicsIntegrationGroup](#) (JeodIntegrationGroupOwner &owner, er7_utils::IntegratorConstructor &integ_cotr, JeodIntegratorInterface &integ_inter, JeodIntegrationTime &time_mngr)
DynamicsIntegrationGroup non-default constructor, used to create the default integration group.
- virtual [~DynamicsIntegrationGroup](#) ()
DynamicsIntegrationGroup destructor.
- bool [is_empty](#) (void) const

Query whether the group is void of registered bodies.
- virtual [DynamicsIntegrationGroup * create_group](#) (JeodIntegrationGroupOwner &owner, er7_utils::IntegratorConstructor &integ_cotr, JeodIntegratorInterface &integ_inter, JeodIntegrationTime &time_mngr) const
Create an integration group object that can be used as the dynamic manager's default integration group.
- virtual void [register_group](#) (DynManager &dyn_manager)
Pre-initialize the group and register it with the dynamics manager.
- virtual void [initialize_group](#) (DynManager &dyn_manager)
Complete the initialization of the group.
- virtual void [prepare_for_integ_loop](#) (double sim_endtime)
Perform actions that need to be taken before entering the derivative / integration loop.
- virtual void [gravitation](#) (DynManager &dyn_manager, GravityManager &gravity_manager)
Compute the gravitational acceleration of each root dynamic body.
- virtual void [collect_derivatives](#) (void)
Collect the forces and torques acting on each root dynamic body.
- virtual er7_utils::IntegratorResult [integrate_bodies](#) (double cycle_dyndt, unsigned int target_stage)
Integrate the states of the DynBody objects that comprise the group.
- virtual void [add_dyn_body](#) (DynBody &body)
Add a DynBody to the set of bodies whose states are integrated by this group.
- virtual void [delete_dyn_body](#) (DynBody &body)
Remove a DynBody from the set of bodies whose states are integrated by this group.

Data Fields

- bool [deriv_ephem_update](#)
Update ephemerides at the derivative rate?

Protected Member Functions

- virtual void [reset_body_integrators](#) (void)
Force all integrators to reset themselves.

Protected Attributes

- JeodPointerVector< DynBody >::type [dyn_bodies](#)
List of vehicles whose state is integrated by this group.
- bool [bodies_integrated_separately](#)
This flag is always true for JEOD integration groups.

Private Member Functions

- void [register_base_contents](#) (void)
Register types and containers.
- [DynamicsIntegrationGroup](#) (const [DynamicsIntegrationGroup](#) &)
Not implemented.
- [DynamicsIntegrationGroup](#) & [operator=](#) (const [DynamicsIntegrationGroup](#) &)
Not implemented.

Friends

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

8.2.1 Detailed Description

A [DynamicsIntegrationGroup](#) integrates the state of a set of DynBoby objects over time.

The class provides implementations of all virtual functions listed below and the pure virtuals defined in the base class. This class is designed for extensibility. Authors of derived classes should follow the extension notes in the source file.

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

8.2.2 Constructor & Destructor Documentation

8.2.2.1 DynamicsIntegrationGroup() [1/3]

```
jeod::DynamicsIntegrationGroup::DynamicsIntegrationGroup ( )
```

[DynamicsIntegrationGroup](#) default constructor, needed for checkpoint/restart.

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

References `register_base_contents()`.

8.2.2.2 DynamicsIntegrationGroup() [2/3]

```
jeod::DynamicsIntegrationGroup::DynamicsIntegrationGroup (
    JeodIntegrationGroupOwner & owner,
    er7_utils::IntegratorConstructor & integ_cotr,
    JeodIntegratorInterface & integ_inter,
    JeodIntegrationTime & time_mngr ) [explicit]
```

[DynamicsIntegrationGroup](#) non-default constructor, used to create the default integration group.

Parameters

in	<i>owner</i>	The new group's owner
in	<i>integ_cotr</i>	Integrator constructor
in	<i>integ_inter</i>	Simulation engine integration interface
in	<i>time_mngr</i>	Time manager

Definition at line 74 of file `dynamics_integration_group.cc`.

References `register_base_contents()`.

8.2.2.3 ~DynamicsIntegrationGroup()

```
jeod::DynamicsIntegrationGroup::~DynamicsIntegrationGroup ( ) [virtual]
```

[DynamicsIntegrationGroup](#) destructor.

Definition at line 103 of file `dynamics_integration_group.cc`.

References `dyn_bodies`.

8.2.2.4 DynamicsIntegrationGroup() [3/3]

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

Not implemented.

8.2.3 Member Function Documentation

8.2.3.1 add_dyn_body()

```
void jeod::DynamicsIntegrationGroup::add_dyn_body (
    DynBody & dyn_body ) [virtual]
```

Add a DynBody to the set of bodies whose states are integrated by this group.

Parameters

<i>dyn_body</i>	DynBody to be added to the group.
-----------------	-----------------------------------

Definition at line 191 of file dynamics_integration_group.cc.

References bodies_integrated_separately, jeod::DynManagerMessages::duplicate_entry, and dyn_bodies.

Referenced by jeod::DynManager::update_integration_group().

8.2.3.2 collect_derivatives()

```
void jeod::DynamicsIntegrationGroup::collect_derivatives (
    void ) [virtual]
```

Collect the forces and torques acting on each root dynamic body.

Definition at line 334 of file dynamics_integration_group.cc.

References dyn_bodies.

Referenced by jeod::DynManager::compute_derivatives().

8.2.3.3 create_group()

```
DynamicsIntegrationGroup * jeod::DynamicsIntegrationGroup::create_group (
    JeodIntegrationGroupOwner & owner,
    er7_utils::IntegratorConstructor & integ_cotr,
    JeodIntegratorInterface & integ_inter,
    JeodIntegrationTime & time_mngr ) const [virtual]
```

Create an integration group object that can be used as the dynamic manager's default integration group.

Parameters

in	<i>owner</i>	The new group's owner
in	<i>integ_cotr</i>	Integrator constructor
in	<i>integ_inter</i>	Simulation engine integration interface
in	<i>time_mngr</i>	Time manager

Returns

Created [DynamicsIntegrationGroup](#).

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

Referenced by `jeod::DynManager::initialize_model_internal()`.

8.2.3.4 delete_dyn_body()

```
void jeod::DynamicsIntegrationGroup::delete_dyn_body (
    DynBody & dyn_body ) [virtual]
```

Remove a DynBody from the set of bodies whose states are integrated by this group.

Parameters

<i>dyn_body</i>	DynBody to be removed from the group.
-----------------	---------------------------------------

Definition at line 250 of file `dynamics_integration_group.cc`.

References `dyn_bodies`, and `jeod::DynManagerMessages::inconsistent_setup`.

8.2.3.5 gravitation()

```
void jeod::DynamicsIntegrationGroup::gravitation (
    DynManager & dyn_manager,
    GravityManager & gravity_manager ) [virtual]
```

Compute the gravitational acceleration of each root dynamic body.

Parameters

<i>dyn_manager</i>	Dynamics manager.
<i>gravity_manager</i>	Gravity Manager.

Definition at line 301 of file `dynamics_integration_group.cc`.

References `deriv_ephem_update`, `dyn_bodies`, and `jeod::DynManager::gravitation()`.

Referenced by `jeod::DynManager::gravitation()`.

8.2.3.6 initialize_group()

```
void jeod::DynamicsIntegrationGroup::initialize_group (
    DynManager & dyn_manager ) [virtual]
```

Complete the initialization of the group.

For overrides: This function is called by `DynManager::initialize_simulation`. At the point of this call, the `dyn_bodies` vector is populated with the bodies that are to be integrated by this group. Note well: That vector can still be empty.

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

References `bodies_integrated_separately`, `dyn_bodies`, and `jeod::DynManagerMessages::null_pointer`.

Referenced by `jeod::DynManager::initialize_integ_groups()`.

8.2.3.7 integrate_bodies()

```
er7_utils::IntegratorResult jeod::DynamicsIntegrationGroup::integrate_bodies (
    double cycle_dyndt,
    unsigned int target_stage ) [virtual]
```

Integrate the states of the `DynBody` objects that comprise the group.

Parameters

in	<i>cycle_dyndt</i>	Dynamic time step, in dynamic time seconds.
in	<i>target_stage</i>	The stage of the integration process that the integrator should try to attain.

Returns

The status (time advance, pass/fail status) of the integration.

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

References `bodies_integrated_separately`, `dyn_bodies`, and `jeod::DynManagerMessages::inconsistent_setup`.

8.2.3.8 is_empty()

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

Query whether the group is void of registered bodies.

Returns

True if group is empty, false otherwise.

Definition at line 132 of file `dynamics_integration_group.hh`.

References `dyn_bodies`.

8.2.3.9 operator=()

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

Not implemented.

8.2.3.10 prepare_for_integ_loop()

```
void jeod::DynamicsIntegrationGroup::prepare_for_integ_loop (
    double sim_endtime ) [virtual]
```

Perform actions that need to be taken before entering the derivative / integration loop.

The base action is to set the time model to the time at the start of the integration loop.

Parameters

<i>sim_endtime</i>	End time of integration loop.
--------------------	-------------------------------

Definition at line 288 of file `dynamics_integration_group.cc`.

8.2.3.11 register_base_contents()

```
void jeod::DynamicsIntegrationGroup::register_base_contents (
    void ) [private]
```

Register types and containers.

Definition at line 93 of file `dynamics_integration_group.cc`.

References `dyn_bodies`.

Referenced by `DynamicsIntegrationGroup()`.

8.2.3.12 register_group()

```
void jeod::DynamicsIntegrationGroup::register_group (
    DynManager & dyn_manager ) [virtual]
```

Pre-initialize the group and register it with the dynamics manager.

This function is to be called early in the initialization process. Overrides should not depend on the dyn_bodies vector having any members.

Parameters

in	<i>dyn_manager</i>	Dynamics manager.
----	--------------------	-------------------

Definition at line 139 of file dynamics_integration_group.cc.

References jeod::DynManager::add_integ_group(), and jeod::DynManager::is_integ_group_registered().

8.2.3.13 reset_body_integrators()

```
void jeod::DynamicsIntegrationGroup::reset_body_integrators (
    void ) [protected], [virtual]
```

Force all integrators to reset themselves.

Definition at line 358 of file dynamics_integration_group.cc.

References dyn_bodies.

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

```
void init_attrjeod__DynamicsIntegrationGroup ( ) [friend]
```

8.2.4.2 InputProcessor

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

Definition at line 92 of file dynamics_integration_group.hh.

8.2.5 Field Documentation

8.2.5.1 `bodies_integrated_separately`

```
bool jeod::DynamicsIntegrationGroup::bodies_integrated_separately [protected]
```

This flag is always true for JEOD integration groups.

Setting this flag to false results in bypassing the call in [DynamicsIntegrationGroup::add_dyn_body](#) to [DynBody::create_body_integrators](#). This hook exists for derived classes that override [DynamicsIntegrationGroup::integrate_bodies](#) in a way that does not involve calling [DynBody::integrate.trick_units\(-\)](#)

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

Referenced by [add_dyn_body\(\)](#), [initialize_group\(\)](#), and [integrate_bodies\(\)](#).

8.2.5.2 `deriv_ephem_update`

```
bool jeod::DynamicsIntegrationGroup::deriv_ephem_update
```

Update ephemerides at the derivative rate?

[trick_units\(-\)](#)

Definition at line 207 of file `dynamics_integration_group.hh`.

Referenced by [jeod::DynManager::gravitation\(\)](#), and [gravitation\(\)](#).

8.2.5.3 `dyn_bodies`

```
JeodPointerVector<DynBody>::type jeod::DynamicsIntegrationGroup::dyn_bodies [protected]
```

List of vehicles whose state is integrated by this group.

[trick_io\(**\)](#)

Definition at line 228 of file `dynamics_integration_group.hh`.

Referenced by [add_dyn_body\(\)](#), [collect_derivatives\(\)](#), [delete_dyn_body\(\)](#), [gravitation\(\)](#), [initialize_group\(\)](#), [integrate_bodies\(\)](#), [is_empty\(\)](#), [register_base_contents\(\)](#), [reset_body_integrators\(\)](#), and [~DynamicsIntegrationGroup\(\)](#).

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

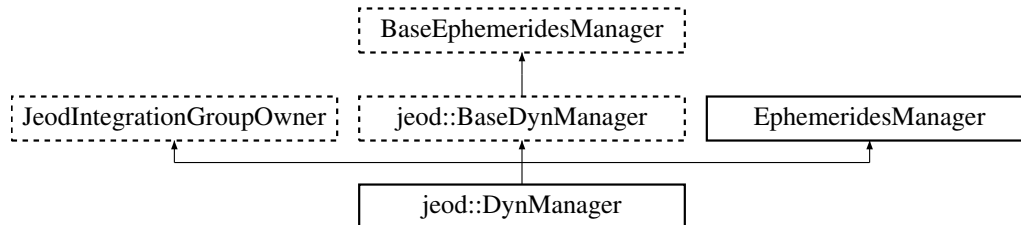
- [dynamics_integration_group.hh](#)
- [dynamics_integration_group.cc](#)

8.3 jeod::DynManager Class Reference

The [DynManager](#) class manages the dynamic elements of a simulation.

```
#include <dyn_manager.hh>
```

Inheritance diagram for jeod::DynManager:



Public Member Functions

- [DynManager](#) ()
DynManager default constructor.
- virtual [~DynManager](#) ()
DynManager destructor.
- bool [is_initialized](#) ()
Determine if the manager has been initialized.
- void [initialize_model](#) ([DynManagerInit](#) &init, TimeManager &time_mgr)
Begin initialization of the JEOD manager model.
- void [initialize_model](#) (JeodIntegratorInterface &integ_if, [DynManagerInit](#) &init, TimeManager &time_mgr)
Begin initialization of the JEOD manager model.
- void [initialize_simulation](#) (void)
Complete initialization of the JEOD manager model.
- virtual void [set_gravity_manager](#) (GravityManager &gravity)
Set the Gravity Manager to the specified reference.
- virtual void [initialize_gravity_controls](#) (void)
Initialize the gravity controls for each dynamic body.
- virtual void [reset_gravity_controls](#) (void)
Reset the gravity controls for each dynamic body.
- void [gravitation](#) (void)
Compute gravitational acceleration on each root body.
- virtual void [add_mass_body](#) (MassBody &mass_body)
Add a mass body to the mass body registry.
- virtual void [add_mass_body](#) (MassBody *mass_body)
Add a mass body to the mass body registry.
- virtual MassBody * [find_mass_body](#) (const char *name) const
Find the mass body with the given name.
- virtual bool [is_mass_body_registered](#) (const MassBody *mass_body) const
Determine if the specified body has been registered with the [DynManager](#).
- virtual void [add_dyn_body](#) (DynBody &dyn_body)
Add a dynamic body to the dynamic body registry.
- virtual DynBody * [find_dyn_body](#) (const char *name) const

- Find the dynamic body with the given name.*

 - virtual std::vector< DynBody * > [get_dyn_bodies](#) () const

Return a copy of the list of registered dynamic bodies.
- virtual bool [is_dyn_body_registered](#) (const DynBody *dyn_body) const

Determine if the specified body has been registered with the [DynManager](#).
- virtual void [add_integ_group](#) (DynamicsIntegrationGroup &integ_group)

Add an integration group to the integration group registry.
- virtual bool [is_integ_group_registered](#) (const DynamicsIntegrationGroup *integ_group) const

Determine if the specified group has been registered with the [DynManager](#).
- void [add_body_action](#) (BodyAction *body_action)

Add a body action to the list of such.
- void [remove_body_action](#) (char *action_name_in)

Remove a body action to the list of such.
- void [perform_actions](#) (void)

Perform dynamic body actions that are ready to be applied.
- void [initialize_integ_groups](#) (void)

Complete initialization of the initialization groups.
- virtual void [update_integration_group](#) (JeodIntegrationGroup &group)

Add DynBody objects to the default integration group.
- void [initialize_dyn_bodies](#) (void)

Initialize dynamic bodies.
- void [initialize_dyn_body](#) (DynBody &body)

Initialize a specific dynamic body.
- void [compute_derivatives](#) ()

Collect forces and torques on each body and compute derivatives.
- virtual void [reset_integrators](#) ()

Force all integrators to reset themselves.
- virtual void [reset_integrators](#) (DynamicsIntegrationGroup &integ_group)

Instruct specific integrator to reset itself.
- int [integrate](#) (double to_sim_time, TimeManager &)

Propagate all vehicles and propagate time.
- virtual double [timestamp](#) (void) const

Return last update time.
- const char * [name](#) (void) const

Return identifier.
- void [shutdown](#) (void)

Shutdown the manager.

Data Fields

- bool [deriv_ephem_update](#)

Update ephemerides at the derivative rate?
- bool [gravity_off](#)

This flag exists primarily to support unit tests.
- [DynManagerInit::EphemerisMode mode](#)

The ephemeris mode in which the dynamics manager operates.
- Trick::Integrator * [sim_integrator](#)

Pointer to the integration object used by the simulation engine itself.

Protected Member Functions

- virtual void [initialize_model_internal](#) (DynManagerInit &init, TimeManager &time_mgr)
Begin initialization of the JEOD manager model.
- void [perform_mass_body_initializations](#) (MassBody *body=NULL)
Initialize all queued body actions that derive from MassBodyInit and apply those that are immediately ready to be applied.
- void [perform_mass_attach_initializations](#) (void)
Initialize all queued body actions that derive from MassBodyAttach and apply those that are immediately ready to be applied.
- void [perform_dyn_body_initializations](#) (DynBody *body=NULL)
Initialize dynamic bodies.
- void [check_for_uninitialized_states](#) (void)
Ensure that all of the required states have been set.

Protected Attributes

- bool [initialized](#)
Have all initializations been performed?
- GravityManager * [gravity_manager](#)
The model that encapsulates all of the gravity models.
- er7_utils::IntegratorConstructor * [integ_constructor](#)
Integrator generator.
- JeodIntegratorInterface * [integ_interface](#)
Interface with the simulation integration structure.
- DynamicsIntegrationGroup * [default_integ_group](#)
The integration group used for simple monolithic simulations.
- SinglePointEphemeris * [simple_ephemeris](#)
Simple ephemeris for use in empty space and single planet modes.
- std::vector< MassBody * > [mass_bodies](#)
List of vehicle models.
- std::vector< DynBody * > [dyn_bodies](#)
List of vehicle models.
- std::vector< DynamicsIntegrationGroup * > [integ_groups](#)
List of integration groups.
- std::list< BodyAction * > [body_actions](#)
List of body initializers.

Private Member Functions

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

Friends

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

8.3.1 Detailed Description

The [DynManager](#) class manages the dynamic elements of a simulation.

The primary functions of a [DynManager](#) are to:

- Dynamically determine which ephemerides are needed in a simulation.
- Initialize ephemeris models and keep them in sync with the rest of the simulation.
- Initialize mass bodies and dynamic bodies independently of the order in which these bodies are declared in the S_define file.
- Coordinate the computation of the cumulative forces and torques and gravitational effects on the dynamic bodies in a simulation.
- Coordinate the integration of time and of dynamic body states.
- Apply asynchronous actions to bodies.

The [DynManager](#) can operate in one of three modes: empty space, single planet, and ephemeris mode. The [DynManager](#) inherits from EphemerisInterface so that when it operates in empty space or single-planet mode it can properly register itself as the owner of the reference frame tree root node.

Definition at line 115 of file dyn_manager.hh.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 DynManager() [1/2]

```
jeod::DynManager::DynManager (
    void )
```

[DynManager](#) default constructor.

Definition at line 66 of file dyn_manager.cc.

8.3.2.2 ~DynManager()

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

[DynManager](#) destructor.

Definition at line 101 of file dyn_manager.cc.

References [default_integ_group](#), [integ_constructor](#), [integ_interface](#), and [simple_ephemeris](#).

8.3.2.3 DynManager() [2/2]

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

Not implemented.

8.3.3 Member Function Documentation

8.3.3.1 add_body_action()

```
void jeod::DynManager::add_body_action (
    BodyAction * body_action )
```

Add a body action to the list of such.

Parameters

<i>in, out</i>	<i>body_action</i>	Body action
----------------	--------------------	-------------

Definition at line 207 of file `dyn_manager.cc`.

References `body_actions`, `jeod::DynManagerMessages::duplicate_entry`, `initialized`, and `jeod::DynManagerMessages::null_pointer`.

8.3.3.2 add_dyn_body()

```
void jeod::DynManager::add_dyn_body (
    DynBody & dyn_body ) [virtual]
```

Add a dynamic body to the dynamic body registry.

Parameters

<i>dyn_body</i>	Dynamic body to be added to the registry.
-----------------	---

Implements [jeod::BaseDynManager](#).

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

References `add_mass_body()`, `jeod::DynManagerMessages::duplicate_entry`, `dyn_bodies`, `find_dyn_body()`, `find_mass_body()`, `jeod::DynManagerMessages::invalid_name`, and `is_dyn_body_registered()`.

8.3.3.3 add_integ_group()

```
void jeod::DynManager::add_integ_group (
    DynamicsIntegrationGroup & integ_group ) [virtual]
```

Add an integration group to the integration group registry.

Parameters

<i>integ_group</i>	Integration group to be added.
--------------------	--------------------------------

Implements [jeod::BaseDynManager](#).

Definition at line 69 of file integ_group_primitives.cc.

References [default_integ_group](#), [jeod::DynManagerMessages::duplicate_entry](#), [jeod::DynManagerMessages::inconsistent_setup](#), [initialized](#), [integ_groups](#), and [is_integ_group_registered\(\)](#).

Referenced by [jeod::DynamicsIntegrationGroup::register_group\(\)](#).

8.3.3.4 add_mass_body() [1/2]

```
void jeod::DynManager::add_mass_body (
    MassBody & mass_body ) [virtual]
```

Add a mass body to the mass body registry.

Parameters

<i>mass_body</i>	Mass body to be added to the registry.
------------------	--

Implements [jeod::BaseDynManager](#).

Definition at line 98 of file mass_bodies_primitives.cc.

References [jeod::DynManagerMessages::duplicate_entry](#), [find_mass_body\(\)](#), [is_mass_body_registered\(\)](#), and [mass_bodies](#).

Referenced by [add_dyn_body\(\)](#), and [add_mass_body\(\)](#).

8.3.3.5 add_mass_body() [2/2]

```
void jeod::DynManager::add_mass_body (
    MassBody * mass_body ) [virtual]
```

Add a mass body to the mass body registry.

Parameters

<i>mass_body</i>	Mass body to be added to the registry.
------------------	--

Implements [jeod::BaseDynManager](#).

Definition at line 137 of file `mass_bodies_primitives.cc`.

References `add_mass_body()`, and `jeod::DynManagerMessages::null_pointer`.

8.3.3.6 `check_for_uninitialized_states()`

```
void jeod::DynManager::check_for_uninitialized_states (
    void ) [protected]
```

Ensure that all of the required states have been set.

Definition at line 366 of file `initialize_dyn_bodies.cc`.

References `dyn_bodies`, and `jeod::DynManagerMessages::inconsistent_setup`.

Referenced by `initialize_dyn_bodies()`.

8.3.3.7 `compute_derivatives()`

```
void jeod::DynManager::compute_derivatives ( ) [inline]
```

Collect forces and torques on each body and compute derivatives.

Definition at line 234 of file `dyn_manager.hh`.

References `jeod::DynamicsIntegrationGroup::collect_derivatives()`, and `default_integ_group`.

8.3.3.8 `find_dyn_body()`

```
DynBody * jeod::DynManager::find_dyn_body (
    const char * body_name ) const [virtual]
```

Find the dynamic body with the given name.

Parameters

<i>body_name</i>	Dynamic body name
------------------	-------------------

Returns

Pointer to found DynBody; NULL if not found.

Implements [jeod::BaseDynManager](#).

Definition at line 57 of file dyn_bodies_primitives.cc.

References dyn_bodies.

Referenced by add_dyn_body().

8.3.3.9 find_mass_body()

```
MassBody * jeod::DynManager::find_mass_body (
    const char * body_name ) const [virtual]
```

Find the mass body with the given name.

Parameters

<i>body_name</i>	Mass body name
------------------	----------------

Returns

Pointer to found MassBody; NULL if not found.

Implements [jeod::BaseDynManager](#).

Definition at line 51 of file mass_bodies_primitives.cc.

References mass_bodies.

Referenced by add_dyn_body(), and add_mass_body().

8.3.3.10 get_dyn_bodies()

```
virtual std::vector<DynBody*> jeod::DynManager::get_dyn_bodies ( ) const [inline], [virtual]
```

Return a copy of the list of registered dynamic bodies.

Returns

Copy of dyn_bodies data member

Implements [jeod::BaseDynManager](#).

Definition at line 190 of file dyn_manager.hh.

References dyn_bodies.

8.3.3.11 gravitation()

```
void jeod::DynManager::gravitation (
    void )
```

Compute gravitational acceleration on each root body.

Definition at line 125 of file gravitation.cc.

References `default_integ_group`, `jeod::DynamicsIntegrationGroup::deriv_ephem_update`, `deriv_ephem_update`, `jeod::DynamicsIntegrationGroup::gravitation()`, `gravity_manager`, `gravity_off`, `jeod::DynManagerMessages::inconsistent_setup`, and `initialized`.

Referenced by `jeod::DynamicsIntegrationGroup::gravitation()`.

8.3.3.12 initialize_dyn_bodies()

```
void jeod::DynManager::initialize_dyn_bodies (
    void )
```

Initialize dynamic bodies.

Definition at line 57 of file initialize_dyn_bodies.cc.

References `body_actions`, `check_for_uninitialized_states()`, `dyn_bodies`, `perform_dyn_body_initializations()`, `perform_mass_attach_initializations()`, and `perform_mass_body_initializations()`.

Referenced by `initialize_simulation()`.

8.3.3.13 initialize_dyn_body()

```
void jeod::DynManager::initialize_dyn_body (
    DynBody & body )
```

Initialize a specific dynamic body.

Assumptions and Limitations

- The body in question is assumed to be an isolated body.

Parameters

<i>in, out</i>	<i>body</i>	Body to be initialized
----------------	-------------	------------------------

Definition at line 109 of file initialize_dyn_bodies.cc.

References `perform_dyn_body_initializations()`, and `perform_mass_body_initializations()`.

8.3.3.14 initialize_gravity_controls()

```
void jeod::DynManager::initialize_gravity_controls (
    void ) [virtual]
```

Initialize the gravity controls for each dynamic body.

Assumptions and Limitations

- Not called in empty space mode.

Implements [jeod::BaseDynManager](#).

Definition at line 51 of file gravitation.cc.

References `dyn_bodies`, `gravity_manager`, `gravity_off`, and `jeod::DynManagerMessages::inconsistent_setup`.

Referenced by `initialize_simulation()`.

8.3.3.15 initialize_integ_groups()

```
void jeod::DynManager::initialize_integ_groups (
    void )
```

Complete initialization of the initialization groups.

Definition at line 108 of file initialize_simulation.cc.

References `default_integ_group`, `jeod::DynamicsIntegrationGroup::initialize_group()`, and `integ_groups`.

Referenced by `initialize_simulation()`.

8.3.3.16 initialize_model() [1/2]

```
void jeod::DynManager::initialize_model (
    DynManagerInit & init,
    TimeManager & time_mgr )
```

Begin initialization of the JEOD manager model.

Parameters

<code>in, out</code>	<code>init</code>	Initialization data
<code>in, out</code>	<code>time_mgr</code>	Time manager

Definition at line 63 of file initialize_model.cc.

References `initialize_model_internal()`, `integ_interface`, and `sim_integrator`.

8.3.3.17 initialize_model() [2/2]

```
void jeod::DynManager::initialize_model (
    JeodIntegratorInterface & integ_if,
    DynManagerInit & init,
    TimeManager & time_mngr )
```

Begin initialization of the JEOD manager model.

Parameters

<i>in, out</i>	<i>integ_if</i>	Integrator interface
<i>in, out</i>	<i>init</i>	Initialization data
<i>in, out</i>	<i>time_mngr</i>	Time manager

Class: (initialization)

Definition at line 84 of file initialize_model.cc.

References `initialize_model_internal()`, `integ_interface`, and `sim_integrator`.

8.3.3.18 initialize_model_internal()

```
void jeod::DynManager::initialize_model_internal (
    DynManagerInit & init,
    TimeManager & time_mngr ) [protected], [virtual]
```

Begin initialization of the JEOD manager model.

Assumptions and Limitations

- The user-input item selection table must have at most one selection rule for a given name. This limitation is an enforced constraint.

Parameters

<i>in, out</i>	<i>init</i>	Initialization data
<i>in, out</i>	<i>time_mngr</i>	Time manager

Definition at line 106 of file initialize_model.cc.

References `jeod::DynManagerInit::central_point_name`, `jeod::DynamicsIntegrationGroup::create_group()`, `default_↵`
`_integ_group`, `jeod::DynManagerInit::EphemerisMode_EmptySpace`, `jeod::DynManagerInit::EphemerisMode_↵`
`Ephemerides`, `jeod::DynManagerInit::EphemerisMode_SinglePlanet`, `jeod::DynManagerMessages::inconsistent_↵`
`setup`, `jeod::DynManagerInit::integ_constructor`, `integ_constructor`, `jeod::DynManagerInit::integ_group_constructor`,
`integ_groups`, `integ_interface`, `jeod::DynManagerMessages::invalid_name`, `jeod::DynManagerInit::jeod_integ_opt`,
`jeod::DynManagerInit::mode`, `mode`, `jeod::DynManagerInit::sim_integ_opt`, and `simple_ephemeris`.

Referenced by `initialize_model()`.

8.3.3.19 initialize_simulation()

```
void jeod::DynManager::initialize_simulation (
    void )
```

Complete initialization of the JEOD manager model.

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

References `jeod::DynManagerInit::EphemerisMode_EmptySpace`, `gravity_manager`, `gravity_off`, `jeod::Dyn↵`
`ManagerMessages::inconsistent_setup`, `initialize_dyn_bodies()`, `initialize_gravity_controls()`, `initialize_integ↵`
`groups()`, `initialized`, and `mode`.

8.3.3.20 integrate()

```
int jeod::DynManager::integrate (
    double to_sim_time,
    TimeManager & ) [inline]
```

Propagate all vehicles and propagate time.

Parameters

<code>to_sim_time</code>	Simulation time seconds of end of integration interval.
--------------------------	---

Returns

zero if complete, non-zero if incomplete.

Definition at line 258 of file `dyn_manager.hh`.

References `default_integ_group`.

8.3.3.21 is_dyn_body_registered()

```
bool jeod::DynManager::is_dyn_body_registered (
    const DynBody * dyn_body ) const [virtual]
```

Determine if the specified body has been registered with the [DynManager](#).

Parameters

<i>dyn_body</i>	Dynamic body to be found.
-----------------	---------------------------

Returns

True if body has been registered, false otherwise.

Implements [jeod::BaseDynManager](#).

Definition at line 90 of file `dyn_bodies_primitives.cc`.

References `dyn_bodies`.

Referenced by `add_dyn_body()`.

8.3.3.22 `is_initialized()`

```
bool jeod::DynManager::is_initialized ( ) [inline]
```

Determine if the manager has been initialized.

Returns

Initialization status

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

References `initialized`.

8.3.3.23 `is_integ_group_registered()`

```
bool jeod::DynManager::is_integ_group_registered (
    const DynamicsIntegrationGroup * integ_group ) const [virtual]
```

Determine if the specified group has been registered with the [DynManager](#).

Parameters

<i>integ_group</i>	Integration group to be found.
--------------------	--------------------------------

Returns

True if `integ_group` has been registered, false otherwise.

Implements [jeod::BaseDynManager](#).

Definition at line 55 of file integ_group_primitives.cc.

References integ_groups.

Referenced by add_integ_group(), and jeod::DynamicsIntegrationGroup::register_group().

8.3.3.24 is_mass_body_registered()

```
bool jeod::DynManager::is_mass_body_registered (
    const MassBody * mass_body ) const [virtual]
```

Determine if the specified body has been registered with the [DynManager](#).

Parameters

<i>mass_body</i>	Mass body to be found.
------------------	------------------------

Returns

True if body has been registered, false otherwise.

Implements [jeod::BaseDynManager](#).

Definition at line 84 of file mass_bodies_primitives.cc.

References mass_bodies.

Referenced by add_mass_body().

8.3.3.25 name()

```
const char * jeod::DynManager::name (
    void ) const
```

Return identifier.

Returns

Name

Definition at line 143 of file dyn_manager.cc.

8.3.3.26 operator=()

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

Not implemented.

8.3.3.27 perform_actions()

```
void jeod::DynManager::perform_actions (
    void )
```

Perform dynamic body actions that are ready to be applied.

Definition at line 44 of file perform_actions.cc.

References body_actions.

8.3.3.28 perform_dyn_body_initializations()

```
void jeod::DynManager::perform_dyn_body_initializations (
    DynBody * body = NULL ) [protected]
```

Initialize dynamic bodies.

Parameters

in, out	body	Body to be initialized
---------	------	------------------------

Definition at line 243 of file initialize_dyn_bodies.cc.

References body_actions, and jeod::DynManagerMessages::inconsistent_setup.

Referenced by initialize_dyn_bodies(), and initialize_dyn_body().

8.3.3.29 perform_mass_attach_initializations()

```
void jeod::DynManager::perform_mass_attach_initializations (
    void ) [protected]
```

Initialize all queued body actions that derive from MassBodyAttach and apply those that are immediately ready to be applied.

Definition at line 190 of file initialize_dyn_bodies.cc.

References body_actions.

Referenced by initialize_dyn_bodies().

8.3.3.30 perform_mass_body_initializations()

```
void jeod::DynManager::perform_mass_body_initializations (
    MassBody * body = NULL ) [protected]
```

Initialize all queued body actions that derive from MassBodyInit and apply those that are immediately ready to be applied.

Parameters

in, out	<i>body</i>	Body to be initialized
---------	-------------	------------------------

Definition at line 130 of file initialize_dyn_bodies.cc.

References `body_actions`.

Referenced by `initialize_dyn_bodies()`, and `initialize_dyn_body()`.

8.3.3.31 remove_body_action()

```
void jeod::DynManager::remove_body_action (
    char * action_name_in )
```

Remove a body action to the list of such.

Parameters

in	<i>action_name_in</i>	Name of the action to remove
----	-----------------------	------------------------------

Definition at line 253 of file dyn_manager.cc.

References `body_actions`.

8.3.3.32 reset_gravity_controls()

```
void jeod::DynManager::reset_gravity_controls (
    void ) [virtual]
```

Reset the gravity controls for each dynamic body.

Assumptions and Limitations

- Not called in empty space mode.

Implements [jeod::BaseDynManager](#).

Definition at line 86 of file gravitation.cc.

References `dyn_bodies`, `gravity_manager`, `gravity_off`, and `jeod::DynManagerMessages::inconsistent_setup`.

8.3.3.33 reset_integrators() [1/2]

```
void jeod::DynManager::reset_integrators ( ) [virtual]
```

Force all integrators to reset themselves.

Implements [jeod::BaseDynManager](#).

Definition at line 280 of file dyn_manager.cc.

References [default_integ_group](#), and [integ_groups](#).

8.3.3.34 reset_integrators() [2/2]

```
virtual void jeod::DynManager::reset_integrators (
    DynamicsIntegrationGroup & integ_group ) [inline], [virtual]
```

Instruct specific integrator to reset itself.

Parameters

<i>integ_group</i>	Integration group to be reset.
--------------------	--------------------------------

Implements [jeod::BaseDynManager](#).

Definition at line 248 of file dyn_manager.hh.

8.3.3.35 set_gravity_manager()

```
void jeod::DynManager::set_gravity_manager (
    GravityManager & gravity ) [virtual]
```

Set the Gravity Manager to the specified reference.

Parameters

in	<i>gravity</i>	Gravity Manager
----	----------------	-----------------

Implements [jeod::BaseDynManager](#).

Definition at line 167 of file dyn_manager.cc.

References [gravity_manager](#), [gravity_off](#), [jeod::DynManagerMessages::inconsistent_setup](#), [initialized](#), and [jeod::DynManagerMessages::singleton_error](#).

8.3.3.36 shutdown()

```
void jeod::DynManager::shutdown (
    void )
```

Shutdown the manager.

Empty for now.

Definition at line 155 of file dyn_manager.cc.

8.3.3.37 timestamp()

```
double jeod::DynManager::timestamp (
    void ) const [virtual]
```

Return last update time.

Returns

Name

Implements [jeod::BaseDynManager](#).

Definition at line 130 of file dyn_manager.cc.

8.3.3.38 update_integration_group()

```
void jeod::DynManager::update_integration_group (
    JeodIntegrationGroup & group ) [virtual]
```

Add DynBody objects to the default integration group.

Parameters

<i>in, out</i>	<i>group</i>	Group to be updated
----------------	--------------	---------------------

Definition at line 138 of file initialize_simulation.cc.

References [jeod::DynamicsIntegrationGroup::add_dyn_body\(\)](#), [default_integ_group](#), [dyn_bodies](#), and [jeod::DynManagerMessages::inconsistent_setup](#).

8.3.4 Friends And Related Function Documentation

8.3.4.1 init_attrjeod__DynManager

```
void init_attrjeod__DynManager ( ) [friend]
```

8.3.4.2 InputProcessor

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

Definition at line 120 of file dyn_manager.hh.

8.3.5 Field Documentation

8.3.5.1 body_actions

```
std::list<BodyAction*> jeod::DynManager::body_actions [protected]
```

List of body initializers.

Definition at line 367 of file dyn_manager.hh.

Referenced by `add_body_action()`, `initialize_dyn_bodies()`, `perform_actions()`, `perform_dyn_body_initializations()`, `perform_mass_attach_initializations()`, `perform_mass_body_initializations()`, and `remove_body_action()`.

8.3.5.2 default_integ_group

```
DynamicsIntegrationGroup* jeod::DynManager::default_integ_group [protected]
```

The integration group used for simple monolithic simulations.

`trick_units(-)`

Definition at line 342 of file dyn_manager.hh.

Referenced by `add_integ_group()`, `compute_derivatives()`, `gravitation()`, `initialize_integ_groups()`, `initialize_model↔_internal()`, `integrate()`, `reset_integrators()`, `update_integration_group()`, and `~DynManager()`.

8.3.5.3 deriv_ephem_update

```
bool jeod::DynManager::deriv_ephem_update
```

Update ephemerides at the derivative rate?

trick_units(-)

Definition at line 281 of file dyn_manager.hh.

Referenced by gravitation().

8.3.5.4 dyn_bodies

```
std::vector<DynBody*> jeod::DynManager::dyn_bodies [protected]
```

List of vehicle models.

Definition at line 357 of file dyn_manager.hh.

Referenced by add_dyn_body(), check_for_uninitialized_states(), find_dyn_body(), get_dyn_bodies(), initialize_↔_dyn_bodies(), initialize_gravity_controls(), is_dyn_body_registered(), reset_gravity_controls(), and update_↔_integration_group().

8.3.5.5 gravity_manager

```
GravityManager* jeod::DynManager::gravity_manager [protected]
```

The model that encapsulates all of the gravity models.

trick_units(-)

Definition at line 327 of file dyn_manager.hh.

Referenced by gravitation(), initialize_gravity_controls(), initialize_simulation(), reset_gravity_controls(), and set_↔_gravity_manager().

8.3.5.6 gravity_off

```
bool jeod::DynManager::gravity_off
```

This flag exists primarily to support unit tests.

Typical simulations should not set this flag. The intent is to support simulations that use planetary ephemerides but neither need nor have a gravity model.trick_units(-)

Definition at line 288 of file dyn_manager.hh.

Referenced by gravitation(), initialize_gravity_controls(), initialize_simulation(), reset_gravity_controls(), and set_↔_gravity_manager().

8.3.5.7 initialized

```
bool jeod::DynManager::initialized [protected]
```

Have all initializations been performed?

trick_units(—)

Definition at line 322 of file dyn_manager.hh.

Referenced by add_body_action(), add_integ_group(), gravitation(), initialize_simulation(), is_initialized(), and set←_gravity_manager().

8.3.5.8 integ_constructor

```
er7_utils::IntegratorConstructor* jeod::DynManager::integ_constructor [protected]
```

Integrator generator.

trick_units(—)

Definition at line 332 of file dyn_manager.hh.

Referenced by initialize_model_internal(), and ~DynManager().

8.3.5.9 integ_groups

```
std::vector<DynamicsIntegrationGroup*> jeod::DynManager::integ_groups [protected]
```

List of integration groups.

Definition at line 362 of file dyn_manager.hh.

Referenced by add_integ_group(), initialize_integ_groups(), initialize_model_internal(), is_integ_group_←registered(), and reset_integrators().

8.3.5.10 integ_interface

```
JeodIntegratorInterface* jeod::DynManager::integ_interface [protected]
```

Interface with the simulation integration structure.

trick_units(—)

Definition at line 337 of file dyn_manager.hh.

Referenced by initialize_model(), initialize_model_internal(), and ~DynManager().

8.3.5.11 mass_bodies

```
std::vector<MassBody*> jeod::DynManager::mass_bodies [protected]
```

List of vehicle models.

Definition at line 352 of file dyn_manager.hh.

Referenced by `add_mass_body()`, `find_mass_body()`, and `is_mass_body_registered()`.

8.3.5.12 mode

```
DynManagerInit::EphemerisMode jeod::DynManager::mode
```

The ephemeris mode in which the dynamics manager operates.

`trick_units(-)`

Definition at line 293 of file dyn_manager.hh.

Referenced by `initialize_model_internal()`, and `initialize_simulation()`.

8.3.5.13 sim_integrator

```
Trick::Integrator* jeod::DynManager::sim_integrator
```

Pointer to the integration object used by the simulation engine itself.

`trick_units(-)`

Definition at line 298 of file dyn_manager.hh.

Referenced by `initialize_model()`.

8.3.5.14 simple_ephemeris

```
SinglePointEphemeris* jeod::DynManager::simple_ephemeris [protected]
```

Simple ephemeris for use in empty space and single planet modes.

`trick_units(-)`

Definition at line 347 of file dyn_manager.hh.

Referenced by `initialize_model_internal()`, and `~DynManager()`.

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

- [dyn_manager.hh](#)
- [dyn_bodies_primitives.cc](#)
- [dyn_manager.cc](#)
- [gravitation.cc](#)
- [initialize_dyn_bodies.cc](#)
- [initialize_model.cc](#)
- [initialize_simulation.cc](#)
- [integ_group_primitives.cc](#)
- [mass_bodies_primitives.cc](#)
- [perform_actions.cc](#)

8.4 jeod::DynManagerInit Class Reference

This class contains data used to initialize a [DynManager](#) object.

```
#include <dyn_manager_init.hh>
```

Public Types

- enum [EphemerisMode](#) { [EphemerisMode_EmptySpace](#) = 0, [EphemerisMode_SinglePlanet](#) = 1, [EphemerisMode_Ephemerides](#) = 2 }

Identify modes in which the [DynManager](#) can operate.

Public Member Functions

- [DynManagerInit](#) (void)
[DynManagerInit](#) default constructor.
- [~DynManagerInit](#) (void)
[DynManagerInit](#) destructor.

Data Fields

- [EphemerisMode](#) [mode](#)
Dynamics manager mode.
- char * [central_point_name](#)
Name of central point, used when the manager operates in empty space or single planet mode.
- [DynamicsIntegrationGroup](#) * [integ_group_constructor](#)
An integration group object used by the simulation's dynamics manager to create the default integration group.
- er7_utils::IntegratorConstructor * [integ_constructor](#)
The simulation's dynamics manager uses an integrator constructor to generate the dynamic manager's time integrator and to generate a state integrator for each dynamic body managed by the dynamics manager.
- er7_utils::Integration::Technique [jeod_integ_opt](#)
Integrator type.
- int [sim_integ_opt](#)
Integrator type.

Private Member Functions

- [DynManagerInit](#) (const [DynManagerInit](#) &)
- [DynManagerInit](#) & operator= (const [DynManagerInit](#) &)

8.4.1 Detailed Description

This class contains data used to initialize a [DynManager](#) object.

Definition at line 95 of file [dyn_manager_init.hh](#).

8.4.2 Member Enumeration Documentation

8.4.2.1 EphemerisMode

```
enum jeod::DynManagerInit::EphemerisMode
```

Identify modes in which the [DynManager](#) can operate.

Enumerator

EphemerisMode_EmptySpace	
EphemerisMode_SinglePlanet	
EphemerisMode_Ephemerides	

Definition at line 104 of file dyn_manager_init.hh.

8.4.3 Constructor & Destructor Documentation**8.4.3.1 DynManagerInit()** [1/2]

```
jeod::DynManagerInit::DynManagerInit (
    void )
```

[DynManagerInit](#) default constructor.

Definition at line 46 of file dyn_manager_init.cc.

8.4.3.2 ~DynManagerInit()

```
jeod::DynManagerInit::~~DynManagerInit (
    void )
```

[DynManagerInit](#) destructor.

Definition at line 63 of file dyn_manager_init.cc.

8.4.3.3 DynManagerInit() [2/2]

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

8.4.4 Member Function Documentation**8.4.4.1 operator=()**

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

8.4.5 Field Documentation

8.4.5.1 central_point_name

```
char* jeod::DynManagerInit::central_point_name
```

Name of central point, used when the manager operates in empty space or single planet mode.

trick_units(-)

Definition at line 134 of file dyn_manager_init.hh.

Referenced by jeod::DynManager::initialize_model_internal().

8.4.5.2 integ_constructor

```
er7_utils::IntegratorConstructor* jeod::DynManagerInit::integ_constructor
```

The simulation's dynamics manager uses an integrator constructor to generate the dynamic manager's time integrator and to generate a state integrator for each dynamic body managed by the dynamics manager.

The dynamics manager uses the following priority scheme to identify its integrator constructor:

- The dynamics manager uses the [DynManagerInit](#) integ_constructor data member if that member is not NULL. Note well: This is the only way by which a user-developed integration technique can be used within JEOD.
- The dynamics manager uses the IntegratorConstructorFactory::create method to create an integrator constructor. The value supplied to this method is the first of the following that specifies a valid JEOD integration technique:
- The [DynManagerInit](#) object's jeod_integ_opt data member.
- The JEOD equivalent of the Trick 7 integration structure's option member (Trick 7 only).
- The JEOD equivalent of the [DynManagerInit](#) object's sim_integ_opt data member.trick_units(-)

Definition at line 168 of file dyn_manager_init.hh.

Referenced by jeod::DynManager::initialize_model_internal().

8.4.5.3 integ_group_constructor

```
DynamicsIntegrationGroup* jeod::DynManagerInit::integ_group_constructor
```

An integration group object used by the simulation's dynamics manager to create the default integration group.

The `integ_group_constructor` does not have to be a functional integration group object; it can be created using the group's default constructor. If this object is not NULL, the dynamics manager will call this object's `create_group` method to create a functional integration group object to serve as the simulation's default integration group. If this object is NULL, the dynamics manager will use create the default integration group from the [DynamicsIntegrationGroup](#) class.`trick_units(-)`

Definition at line 147 of file `dyn_manager_init.hh`.

Referenced by `jeod::DynManager::initialize_model_internal()`.

8.4.5.4 jeod_integ_opt

```
er7_utils::Integration::Technique jeod::DynManagerInit::jeod_integ_opt
```

Integrator type.

This data member provides an alternative means for specifying the integration technique to be used. See the `integ_constructor` documentation for `usage.trick_units(-)`

Definition at line 175 of file `dyn_manager_init.hh`.

Referenced by `jeod::DynManager::initialize_model_internal()`.

8.4.5.5 mode

```
EphemerisMode jeod::DynManagerInit::mode
```

Dynamics manager mode.

`trick_units(-)`

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

Referenced by `jeod::DynManager::initialize_model_internal()`.

8.4.5.6 sim_integ_opt

```
int jeod::DynManagerInit::sim_integ_opt
```

Integrator type.

This data member provides yet another alternative means for specifying the integration technique to be used. See the `integ_constructor` documentation for `usage.trick_units(-)`

Definition at line 182 of file `dyn_manager_init.hh`.

Referenced by `jeod::DynManager::initialize_model_internal()`.

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

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

8.5 jeod::DynManagerMessages Class Reference

Specifies the message IDs used in the [DynManager](#) model.

```
#include <dyn_manager_messages.hh>
```

Static Public Attributes

- static char const * [null_pointer](#) = "dynamics/dyn_manager/" "null_pointer"
Issued when a pointer should be non-NULL but isn't.
- static char const * [duplicate_entry](#) = "dynamics/dyn_manager/" "duplicate_entry"
Issued on request to add a pointer to a list a second time.
- static char const * [invalid_name](#) = "dynamics/dyn_manager/" "invalid_name"
Issued when a name is invalid – empty, a duplicate, ...
- static char const * [invalid_frame](#) = "dynamics/dyn_manager/" "invalid_frame"
Issued when a frame is invalid – not an integ frame, ...
- static char const * [invalid_type](#) = "dynamics/dyn_manager/" "invalid_type"
Issued when an object of an unexpected type is encountered.
- static char const * [inconsistent_setup](#) = "dynamics/dyn_manager/" "inconsistent_setup"
Issued when some conditions are inconsistent.
- static char const * [singleton_error](#) = "dynamics/dyn_manager/" "singleton_error"
Error issued when multiple instance of a class that should be a singleton are created or when no such instance exists (but should).
- static char const * [internal_error](#) = "dynamics/dyn_manager/" "internal_error"
Error issued when some internal error occurred.

Private Member Functions

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

Friends

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

8.5.1 Detailed Description

Specifies the message IDs used in the [DynManager](#) model.

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

8.5.2 Constructor & Destructor Documentation

8.5.2.1 `DynManagerMessages()` [1/2]

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

Not implemented.

8.5.2.2 `DynManagerMessages()` [2/2]

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

Not implemented.

8.5.3 Member Function Documentation

8.5.3.1 `operator=()`

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

Not implemented.

8.5.4 Friends And Related Function Documentation

8.5.4.1 init_attrjeod__DynManagerMessages

```
void init_attrjeod__DynManagerMessages ( ) [friend]
```

8.5.4.2 InputProcessor

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

Definition at line 85 of file dyn_manager_messages.hh.

8.5.5 Field Documentation

8.5.5.1 duplicate_entry

```
char const * jeod::DynManagerMessages::duplicate_entry = "dynamics/dyn_manager/" "duplicate_↵  
entry" [static]
```

Issued on request to add a pointer to a list a second time.

trick_units(—)

Definition at line 99 of file dyn_manager_messages.hh.

Referenced by jeod::DynManager::add_body_action(), jeod::DynManager::add_dyn_body(), jeod::Dynamics↵
IntegrationGroup::add_dyn_body(), jeod::DynManager::add_integ_group(), and jeod::DynManager::add_mass_↵
body().

8.5.5.2 inconsistent_setup

```
char const * jeod::DynManagerMessages::inconsistent_setup = "dynamics/dyn_manager/" "inconsistent_↵  
_setup" [static]
```

Issued when some conditions are inconsistent.

trick_units(—)

Definition at line 119 of file dyn_manager_messages.hh.

Referenced by jeod::DynManager::add_integ_group(), jeod::DynManager::check_for_uninitialized_states(), jeod↵
::DynamicsIntegrationGroup::delete_dyn_body(), jeod::DynManager::gravitation(), jeod::DynManager::initialize_↵
gravity_controls(), jeod::DynManager::initialize_model_internal(), jeod::DynManager::initialize_simulation(), jeod↵
::DynamicsIntegrationGroup::integrate_bodies(), jeod::DynManager::perform_dyn_body_initializations(), jeod↵
::DynManager::reset_gravity_controls(), jeod::DynManager::set_gravity_manager(), and jeod::DynManager↵
::update_integration_group().

8.5.5.3 internal_error

```
char const * jeod::DynManagerMessages::internal_error = "dynamics/dyn_manager/" "internal_↵  
error" [static]
```

Error issued when some internal error occurred.

These errors should never happen.
trick_units(-)

Definition at line 131 of file dyn_manager_messages.hh.

8.5.5.4 invalid_frame

```
char const * jeod::DynManagerMessages::invalid_frame = "dynamics/dyn_manager/" "invalid_frame"  
[static]
```

Issued when a frame is invalid – not an integ frame, ...

trick_units(-)

Definition at line 109 of file dyn_manager_messages.hh.

8.5.5.5 invalid_name

```
char const * jeod::DynManagerMessages::invalid_name = "dynamics/dyn_manager/" "invalid_name"  
[static]
```

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

trick_units(-)

Definition at line 104 of file dyn_manager_messages.hh.

Referenced by jeod::DynManager::add_dyn_body(), and jeod::DynManager::initialize_model_internal().

8.5.5.6 invalid_type

```
char const * jeod::DynManagerMessages::invalid_type = "dynamics/dyn_manager/" "invalid_type"  
[static]
```

Issued when an object of an unexpected type is encountered.

trick_units(-)

Definition at line 114 of file dyn_manager_messages.hh.

8.5.5.7 null_pointer

```
char const * jeod::DynManagerMessages::null_pointer = "dynamics/dyn_manager/" "null_pointer"  
[static]
```

Issued when a pointer should be non-NULL but isn't.

trick_units(-)

Definition at line 94 of file dyn_manager_messages.hh.

Referenced by jeod::DynManager::add_body_action(), jeod::DynManager::add_mass_body(), and jeod::DynamicsIntegrationGroup::initialize_group().

8.5.5.8 singleton_error

```
char const * jeod::DynManagerMessages::singleton_error = "dynamics/dyn_manager/" "singleton_↵  
error" [static]
```

Error issued when multiple instance of a class that should be a singleton are created or when no such instance exists (but should).

trick_units(-)

Definition at line 125 of file dyn_manager_messages.hh.

Referenced by jeod::DynManager::set_gravity_manager().

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

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

Chapter 9

File Documentation

9.1 `base_dyn_manager.hh` File Reference

Define the BaseDynManager class, which defines the interfaces to the class DynManager.

```
#include "environment/ephemerides/ephem_manager/include/base_ephem_manager.↵  
hh"  
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::BaseDynManager](#)

The [DynManager](#) class augments the EphemManager with dynamics-related items.

Namespaces

- [jeod](#)

Namespace jeod.

9.1.1 Detailed Description

Define the BaseDynManager class, which defines the interfaces to the class DynManager.

9.2 `class_declarations.hh` File Reference

Forward declarations of classes defined in [dyn_manager.hh](#).

Namespaces

- [jeod](#)

Namespace jeod.

9.2.1 Detailed Description

Forward declarations of classes defined in [dyn_manager.hh](#).

9.3 dyn_bodies_primitives.cc File Reference

Define the DynManager member functions that search through and add elements to the collection of DynBody pointers.

```
#include <algorithm>
#include <cstdint>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.3.1 Detailed Description

Define the DynManager member functions that search through and add elements to the collection of DynBody pointers.

9.4 dyn_manager.cc File Reference

Define simple member functions for the DynManager and related classes.

```
#include <cstdint>
#include "dynamics/body_action/include/body_action.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/mass/include/mass.hh"
#include "environment/ephemerides/ephem_interface/include/simple_ephemerides.↵
hh"
#include "environment/ephemerides/ephem_item/include/ephem_item.hh"
#include "environment/planet/include/planet.hh"
#include "utils/integration/include/jeod_integration_group.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
#include "../include/dynamics_integration_group.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.4.1 Detailed Description

Define simple member functions for the DynManager and related classes.

9.5 dyn_manager.hh File Reference

Define the DynManager class, which manages the planets and vehicles modeled in a JEOD-based simulation.

```
#include <list>
#include <vector>
#include "environment/ephemerides/ephem_manager/include/ephem_manager.hh"
#include "environment/planet/include/planet.hh"
#include "utils/integration/include/jeod_integration_group.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "base_dyn_manager.hh"
#include "dyn_manager_init.hh"
#include "dynamics_integration_group.hh"
#include "environment/ephemerides/ephem_interface/include/simple_ephemerides.↵
hh"
#include "er7_utils/integration/core/include/integrator_constructor_factory.↵
hh"
```

Data Structures

- class [jeod::DynManager](#)
The [DynManager](#) class manages the dynamic elements of a simulation.

Namespaces

- [jeod](#)
Namespace jeod.

9.5.1 Detailed Description

Define the DynManager class, which manages the planets and vehicles modeled in a JEOD-based simulation.

9.6 dyn_manager_init.cc File Reference

Define member functions for the DynManagerInit class.

```
#include <cstddef>
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/dyn_manager_init.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.6.1 Detailed Description

Define member functions for the DynManagerInit class.

9.7 dyn_manager_init.hh File Reference

Define the DynManagerInit class, which contains the data used to initialize a DynManager object.

```
#include "er7_utils/integration/core/include/integration_technique.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class [jeod::DynManagerInit](#)

This class contains data used to initialize a [DynManager](#) object.

Namespaces

- [er7_utils](#)

Namespace [er7_utils](#) contains the state integration models used by JEOD.

- [jeod](#)

Namespace jeod.

9.7.1 Detailed Description

Define the DynManagerInit class, which contains the data used to initialize a DynManager object.

9.8 dyn_manager_messages.cc File Reference

Implement the class DynManagerMessages.

```
#include "utils/message/include/make_message_code.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

Macros

- `#define MAKE_DYNMANAGER_MESSAGE_CODE(id) JEOD_MAKE_MESSAGE_CODE(DynManagerMessages, "dynamics/dyn_manager/", id)`

9.8.1 Detailed Description

Implement the class DynManagerMessages.

9.8.2 Macro Definition Documentation

9.8.2.1 MAKE_DYNMANAGER_MESSAGE_CODE

```
#define MAKE_DYNMANAGER_MESSAGE_CODE(  
    id ) JEOD_MAKE_MESSAGE_CODE(DynManagerMessages, "dynamics/dyn_manager/", id)
```

Definition at line 38 of file dyn_manager_messages.cc.

9.9 dyn_manager_messages.hh File Reference

Define the class DynManagerMessages, the class that specifies the message IDs used in the DynManager model.

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

Data Structures

- class `jeod::DynManagerMessages`
Specifies the message IDs used in the `DynManager` model.

Namespaces

- `jeod`
Namespace jeod.

9.9.1 Detailed Description

Define the class DynManagerMessages, the class that specifies the message IDs used in the DynManager model.

9.10 dynamics_integration_group.cc File Reference

Define DynamicsIntegrationGroup methods.

```
#include <cstdint>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "environment/gravity/include/gravity_manager.hh"
#include "utils/integration/include/jeod_integration_time.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
#include "../include/dynamics_integration_group.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.10.1 Detailed Description

Define DynamicsIntegrationGroup methods.

9.11 dynamics_integration_group.hh File Reference

Define the extensible class DynamicsIntegrationGroup, an instance of which is responsible for integrating the states of a set of DynBody objects.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/container/include/pointer_vector.hh"
#include "utils/integration/include/jeod_integration_group.hh"
```

Data Structures

- class [jeod::DynamicsIntegrationGroup](#)

A [DynamicsIntegrationGroup](#) integrates the state of a set of DynBoby objects over time.

Namespaces

- [jeod](#)

Namespace jeod.

9.11.1 Detailed Description

Define the extensible class `DynamicsIntegrationGroup`, an instance of which is responsible for integrating the states of a set of `DynBody` objects.

9.12 gravitation.cc File Reference

Compute gravitational acceleration.

```
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "environment/gravity/include/gravity_manager.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.12.1 Detailed Description

Compute gravitational acceleration.

9.13 initialize_dyn_bodies.cc File Reference

Define `DynManager::initialize_dyn_bodies`.

```
#include <cstdlib>
#include "dynamics/body_action/include/body_action.hh"
#include "dynamics/body_action/include/body_attach.hh"
#include "dynamics/body_action/include/mass_body_init.hh"
#include "dynamics/body_action/include/dyn_body_init.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/ref_frame_items.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.13.1 Detailed Description

Define DynManager::initialize_dyn_bodies.

9.14 initialize_model.cc File Reference

Define DynManager::initialize_model.

```
#include <cstdint>
#include "er7_utils/integration/core/include/integrator_constructor.hh"
#include "er7_utils/integration/core/include/integrator_constructor_factory.↵
hh"
#include "environment/ephemerides/ephem_interface/include/simple_ephemerides.↵
hh"
#include "environment/ephemerides/ephem_item/include/ephem_item.hh"
#include "environment/time/include/time_manager.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/sim_interface/include/jeod_integrator_interface.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.14.1 Detailed Description

Define DynManager::initialize_model.

9.15 initialize_simulation.cc File Reference

Define DynManager::initialize_simulation, which completes the initialization of the JEOD dynamics manager.

```
#include <cstdint>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "environment/gravity/include/gravity_manager.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)
Namespace jeod.

9.15.1 Detailed Description

Define DynManager::initialize_simulation, which completes the initialization of the JEOD dynamics manager.

9.16 integ_group_primitives.cc File Reference

Define the DynManager member functions that search through and add elements to the collection of Dynamics↵ IntegrationGroup pointers.

```
#include <algorithm>
#include <cstdlib>
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.16.1 Detailed Description

Define the DynManager member functions that search through and add elements to the collection of Dynamics↵ IntegrationGroup pointers.

9.17 mass_bodies_primitives.cc File Reference

Define the DynManager member functions that search through and add elements to the collection of MassBody pointers.

```
#include <algorithm>
#include <cstdlib>
#include "dynamics/mass/include/mass.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_manager.hh"
#include "../include/dyn_manager_messages.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.17.1 Detailed Description

Define the DynManager member functions that search through and add elements to the collection of MassBody pointers.

9.18 perform_actions.cc File Reference

Define DynManager::perform_actions.

```
#include <cstdio>
#include <cstring>
#include "dynamics/body_action/include/body_action.hh"
#include "../include/dyn_manager.hh"
```

Namespaces

- [jeod](#)

Namespace jeod.

9.18.1 Detailed Description

Define DynManager::perform_actions.

Index

- ~BaseDynManager
 - jeod::BaseDynManager, [18](#)
- ~DynManager
 - jeod::DynManager, [36](#)
- ~DynManagerInit
 - jeod::DynManagerInit, [56](#)
- ~DynamicsIntegrationGroup
 - jeod::DynamicsIntegrationGroup, [26](#)
- add_body_action
 - jeod::DynManager, [37](#)
- add_dyn_body
 - jeod::BaseDynManager, [18](#)
 - jeod::DynManager, [37](#)
 - jeod::DynamicsIntegrationGroup, [27](#)
- add_integ_group
 - jeod::BaseDynManager, [19](#)
 - jeod::DynManager, [37](#)
- add_mass_body
 - jeod::BaseDynManager, [19](#)
 - jeod::DynManager, [38](#)
- base_dyn_manager.hh, [65](#)
- bodies_integrated_separately
 - jeod::DynamicsIntegrationGroup, [32](#)
- body_actions
 - jeod::DynManager, [51](#)
- central_point_name
 - jeod::DynManagerInit, [57](#)
- check_for_uninitialized_states
 - jeod::DynManager, [39](#)
- class_declarations.hh, [65](#)
- collect_derivatives
 - jeod::DynamicsIntegrationGroup, [27](#)
- compute_derivatives
 - jeod::DynManager, [39](#)
- create_group
 - jeod::DynamicsIntegrationGroup, [27](#)
- default_integ_group
 - jeod::DynManager, [51](#)
- delete_dyn_body
 - jeod::DynamicsIntegrationGroup, [28](#)
- deriv_ephem_update
 - jeod::DynManager, [51](#)
 - jeod::DynamicsIntegrationGroup, [32](#)
- duplicate_entry
 - jeod::DynManagerMessages, [61](#)
- dyn_bodies
 - jeod::DynManager, [52](#)
 - jeod::DynamicsIntegrationGroup, [32](#)
- dyn_bodies_primitives.cc, [66](#)
- dyn_manager.cc, [66](#)
- dyn_manager.hh, [67](#)
- dyn_manager_init.cc, [67](#)
- dyn_manager_init.hh, [68](#)
- dyn_manager_messages.cc, [68](#)
- MAKE_DYNMANAGER_MESSAGE_CODE, [69](#)
- dyn_manager_messages.hh, [69](#)
- DynManager, [13](#)
 - jeod::DynManager, [36](#)
- DynManagerInit
 - jeod::DynManagerInit, [56](#)
- DynManagerMessages
 - jeod::DynManagerMessages, [60](#)
- Dynamics, [12](#)
- dynamics_integration_group.cc, [70](#)
- dynamics_integration_group.hh, [70](#)
- DynamicsIntegrationGroup
 - jeod::DynamicsIntegrationGroup, [25, 26](#)
- EphemerisMode
 - jeod::DynManagerInit, [55](#)
- er7_utils, [15](#)
- find_dyn_body
 - jeod::BaseDynManager, [20](#)
 - jeod::DynManager, [39](#)
- find_mass_body
 - jeod::BaseDynManager, [20](#)
 - jeod::DynManager, [40](#)
- get_dyn_bodies
 - jeod::BaseDynManager, [20](#)
 - jeod::DynManager, [40](#)
- gravitation
 - jeod::DynManager, [40](#)
 - jeod::DynamicsIntegrationGroup, [28](#)
- gravitation.cc, [71](#)
- gravity_manager
 - jeod::DynManager, [52](#)
- gravity_off
 - jeod::DynManager, [52](#)
- inconsistent_setup
 - jeod::DynManagerMessages, [61](#)
- init_attrjeod__BaseDynManager
 - jeod::BaseDynManager, [23](#)
- init_attrjeod__DynManager

- jeod::DynManager, 50
- init_attrjeod__DynManagerMessages
 - jeod::DynManagerMessages, 60
- init_attrjeod__DynamicsIntegrationGroup
 - jeod::DynamicsIntegrationGroup, 31
- initialize_dyn_bodies
 - jeod::DynManager, 41
- initialize_dyn_bodies.cc, 71
- initialize_dyn_body
 - jeod::DynManager, 41
- initialize_gravity_controls
 - jeod::BaseDynManager, 21
 - jeod::DynManager, 42
- initialize_group
 - jeod::DynamicsIntegrationGroup, 29
- initialize_integ_groups
 - jeod::DynManager, 42
- initialize_model
 - jeod::DynManager, 42, 43
- initialize_model.cc, 72
- initialize_model_internal
 - jeod::DynManager, 43
- initialize_simulation
 - jeod::DynManager, 44
- initialize_simulation.cc, 72
- initialized
 - jeod::DynManager, 52
- InputProcessor
 - jeod::BaseDynManager, 23
 - jeod::DynManager, 51
 - jeod::DynManagerMessages, 61
 - jeod::DynamicsIntegrationGroup, 31
- integ_constructor
 - jeod::DynManager, 53
 - jeod::DynManagerInit, 57
- integ_group_constructor
 - jeod::DynManagerInit, 57
- integ_group_primitives.cc, 73
- integ_groups
 - jeod::DynManager, 53
- integ_interface
 - jeod::DynManager, 53
- integrate
 - jeod::DynManager, 44
- integrate_bodies
 - jeod::DynamicsIntegrationGroup, 29
- internal_error
 - jeod::DynManagerMessages, 61
- invalid_frame
 - jeod::DynManagerMessages, 62
- invalid_name
 - jeod::DynManagerMessages, 62
- invalid_type
 - jeod::DynManagerMessages, 62
- is_dyn_body_registered
 - jeod::BaseDynManager, 21
 - jeod::DynManager, 44
- is_empty
 - jeod::DynamicsIntegrationGroup, 29
- is_initialized
 - jeod::DynManager, 45
- is_integ_group_registered
 - jeod::BaseDynManager, 21
 - jeod::DynManager, 45
- is_mass_body_registered
 - jeod::BaseDynManager, 22
 - jeod::DynManager, 46
- jeod, 15
- jeod::BaseDynManager, 17
 - ~BaseDynManager, 18
 - add_dyn_body, 18
 - add_integ_group, 19
 - add_mass_body, 19
 - find_dyn_body, 20
 - find_mass_body, 20
 - get_dyn_bodies, 20
 - init_attrjeod__BaseDynManager, 23
 - initialize_gravity_controls, 21
 - InputProcessor, 23
 - is_dyn_body_registered, 21
 - is_integ_group_registered, 21
 - is_mass_body_registered, 22
 - reset_gravity_controls, 22
 - reset_integrators, 22
 - set_gravity_manager, 23
 - timestamp, 23
- jeod::DynManager, 33
 - ~DynManager, 36
 - add_body_action, 37
 - add_dyn_body, 37
 - add_integ_group, 37
 - add_mass_body, 38
 - body_actions, 51
 - check_for_uninitialized_states, 39
 - compute_derivatives, 39
 - default_integ_group, 51
 - deriv_ephem_update, 51
 - dyn_bodies, 52
 - DynManager, 36
 - find_dyn_body, 39
 - find_mass_body, 40
 - get_dyn_bodies, 40
 - gravitation, 40
 - gravity_manager, 52
 - gravity_off, 52
 - init_attrjeod__DynManager, 50
 - initialize_dyn_bodies, 41
 - initialize_dyn_body, 41
 - initialize_gravity_controls, 42
 - initialize_integ_groups, 42
 - initialize_model, 42, 43
 - initialize_model_internal, 43
 - initialize_simulation, 44
 - initialized, 52
 - InputProcessor, 51
 - integ_constructor, 53

- integ_groups, 53
- integ_interface, 53
- integrate, 44
- is_dyn_body_registered, 44
- is_initialized, 45
- is_integ_group_registered, 45
- is_mass_body_registered, 46
- mass_bodies, 53
- mode, 54
- name, 46
- operator=, 46
- perform_actions, 47
- perform_dyn_body_initializations, 47
- perform_mass_attach_initializations, 47
- perform_mass_body_initializations, 47
- remove_body_action, 48
- reset_gravity_controls, 48
- reset_integrators, 48, 49
- set_gravity_manager, 49
- shutdown, 49
- sim_integrator, 54
- simple_ephemeris, 54
- timestamp, 50
- update_integration_group, 50
- jeod::DynManagerInit, 55
 - ~DynManagerInit, 56
 - central_point_name, 57
 - DynManagerInit, 56
 - EphemerisMode, 55
 - integ_constructor, 57
 - integ_group_constructor, 57
 - jeod_integ_opt, 58
 - mode, 58
 - operator=, 56
 - sim_integ_opt, 58
- jeod::DynManagerMessages, 59
 - duplicate_entry, 61
 - DynManagerMessages, 60
 - inconsistent_setup, 61
 - init_attrjeod__DynManagerMessages, 60
 - InputProcessor, 61
 - internal_error, 61
 - invalid_frame, 62
 - invalid_name, 62
 - invalid_type, 62
 - null_pointer, 62
 - operator=, 60
 - singleton_error, 63
- jeod::DynamicsIntegrationGroup, 24
 - ~DynamicsIntegrationGroup, 26
 - add_dyn_body, 27
 - bodies_integrated_separately, 32
 - collect_derivatives, 27
 - create_group, 27
 - delete_dyn_body, 28
 - deriv_ephem_update, 32
 - dyn_bodies, 32
 - DynamicsIntegrationGroup, 25, 26
 - gravitation, 28
 - init_attrjeod__DynamicsIntegrationGroup, 31
 - initialize_group, 29
 - InputProcessor, 31
 - integrate_bodies, 29
 - is_empty, 29
 - operator=, 30
 - prepare_for_integ_loop, 30
 - register_base_contents, 30
 - register_group, 30
 - reset_body_integrators, 31
- jeod_integ_opt
 - jeod::DynManagerInit, 58
- MAKE_DYNMANAGER_MESSAGE_CODE
 - dyn_manager_messages.cc, 69
- mass_bodies
 - jeod::DynManager, 53
- mass_bodies_primitives.cc, 73
- mode
 - jeod::DynManager, 54
 - jeod::DynManagerInit, 58
- Models, 11
- name
 - jeod::DynManager, 46
- null_pointer
 - jeod::DynManagerMessages, 62
- operator=
 - jeod::DynManager, 46
 - jeod::DynManagerInit, 56
 - jeod::DynManagerMessages, 60
 - jeod::DynamicsIntegrationGroup, 30
- perform_actions
 - jeod::DynManager, 47
- perform_actions.cc, 74
- perform_dyn_body_initializations
 - jeod::DynManager, 47
- perform_mass_attach_initializations
 - jeod::DynManager, 47
- perform_mass_body_initializations
 - jeod::DynManager, 47
- prepare_for_integ_loop
 - jeod::DynamicsIntegrationGroup, 30
- register_base_contents
 - jeod::DynamicsIntegrationGroup, 30
- register_group
 - jeod::DynamicsIntegrationGroup, 30
- remove_body_action
 - jeod::DynManager, 48
- reset_body_integrators
 - jeod::DynamicsIntegrationGroup, 31
- reset_gravity_controls
 - jeod::BaseDynManager, 22
 - jeod::DynManager, 48
- reset_integrators

- jeod::BaseDynManager, [22](#)
 - jeod::DynManager, [48](#), [49](#)
- set_gravity_manager
 - jeod::BaseDynManager, [23](#)
 - jeod::DynManager, [49](#)
- shutdown
 - jeod::DynManager, [49](#)
- sim_integ_opt
 - jeod::DynManagerInit, [58](#)
- sim_integrator
 - jeod::DynManager, [54](#)
- simple_ephemeris
 - jeod::DynManager, [54](#)
- singleton_error
 - jeod::DynManagerMessages, [63](#)
- timestamp
 - jeod::BaseDynManager, [23](#)
 - jeod::DynManager, [50](#)
- update_integration_group
 - jeod::DynManager, [50](#)