

## DynamicsManagerModel

5.3

Generated by Doxygen 1.8.14



# Contents

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

<b>8 Data Structure Documentation</b>	<b>17</b>
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	26
8.2.2.1 DynamicsIntegrationGroup() [1/3]	26
8.2.2.2 DynamicsIntegrationGroup() [2/3]	26

8.2.2.3	<a href="#">~DynamicsIntegrationGroup()</a> . . . . .	26
8.2.2.4	<a href="#">DynamicsIntegrationGroup()</a> [3/3] . . . . .	27
8.2.3	<a href="#">Member Function Documentation</a> . . . . .	27
8.2.3.1	<a href="#">add_dyn_body()</a> . . . . .	27
8.2.3.2	<a href="#">collect_derivatives()</a> . . . . .	27
8.2.3.3	<a href="#">create_group()</a> . . . . .	27
8.2.3.4	<a href="#">delete_dyn_body()</a> . . . . .	28
8.2.3.5	<a href="#">gravitation()</a> . . . . .	28
8.2.3.6	<a href="#">initialize_group()</a> . . . . .	29
8.2.3.7	<a href="#">integrate_bodies()</a> . . . . .	29
8.2.3.8	<a href="#">is_empty()</a> . . . . .	29
8.2.3.9	<a href="#">operator=()</a> . . . . .	30
8.2.3.10	<a href="#">prepare_for_integ_loop()</a> . . . . .	30
8.2.3.11	<a href="#">register_base_contents()</a> . . . . .	30
8.2.3.12	<a href="#">register_group()</a> . . . . .	30
8.2.3.13	<a href="#">reset_body_integrators()</a> . . . . .	31
8.2.4	<a href="#">Friends And Related Function Documentation</a> . . . . .	31
8.2.4.1	<a href="#">init_attrjeod__DynamicsIntegrationGroup</a> . . . . .	31
8.2.4.2	<a href="#">InputProcessor</a> . . . . .	31
8.2.5	<a href="#">Field Documentation</a> . . . . .	31
8.2.5.1	<a href="#">bodies_integrated_separately</a> . . . . .	32
8.2.5.2	<a href="#">deriv_ephem_update</a> . . . . .	32
8.2.5.3	<a href="#">dyn_bodies</a> . . . . .	32
8.3	<a href="#">jeod::DynManager Class Reference</a> . . . . .	33
8.3.1	<a href="#">Detailed Description</a> . . . . .	36
8.3.2	<a href="#">Constructor &amp; Destructor Documentation</a> . . . . .	36
8.3.2.1	<a href="#">DynManager()</a> [1/2] . . . . .	36
8.3.2.2	<a href="#">~DynManager()</a> . . . . .	36
8.3.2.3	<a href="#">DynManager()</a> [2/2] . . . . .	37
8.3.3	<a href="#">Member Function Documentation</a> . . . . .	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>	45
8.3.3.25	<code>name()</code>	46
8.3.3.26	<code>operator=()</code>	46
8.3.3.27	<code>perform_actions()</code>	46
8.3.3.28	<code>perform_dyn_body_initializations()</code>	46
8.3.3.29	<code>perform_mass_attach_initializations()</code>	47
8.3.3.30	<code>perform_mass_body_initializations()</code>	47

8.3.3.31	<a href="#">remove_body_action()</a>	47
8.3.3.32	<a href="#">reset_gravity_controls()</a>	48
8.3.3.33	<a href="#">reset_integrators()</a> [1/2]	48
8.3.3.34	<a href="#">reset_integrators()</a> [2/2]	48
8.3.3.35	<a href="#">set_gravity_manager()</a>	49
8.3.3.36	<a href="#">shutdown()</a>	49
8.3.3.37	<a href="#">timestamp()</a>	49
8.3.3.38	<a href="#">update_integration_group()</a>	50
8.3.4	<a href="#">Friends And Related Function Documentation</a>	50
8.3.4.1	<a href="#">init_attrjeod__DynManager</a>	50
8.3.4.2	<a href="#">InputProcessor</a>	50
8.3.5	<a href="#">Field Documentation</a>	50
8.3.5.1	<a href="#">body_actions</a>	50
8.3.5.2	<a href="#">default_integ_group</a>	51
8.3.5.3	<a href="#">deriv_ephem_update</a>	51
8.3.5.4	<a href="#">dyn_bodies</a>	51
8.3.5.5	<a href="#">gravity_manager</a>	51
8.3.5.6	<a href="#">gravity_off</a>	52
8.3.5.7	<a href="#">initialized</a>	52
8.3.5.8	<a href="#">integ_constructor</a>	52
8.3.5.9	<a href="#">integ_groups</a>	52
8.3.5.10	<a href="#">integ_interface</a>	53
8.3.5.11	<a href="#">mass_bodies</a>	53
8.3.5.12	<a href="#">mode</a>	53
8.3.5.13	<a href="#">sim_integrator</a>	53
8.3.5.14	<a href="#">simple_ephemeris</a>	54
8.4	<a href="#">jeod::DynManagerInit Class Reference</a>	54
8.4.1	<a href="#">Detailed Description</a>	55
8.4.2	<a href="#">Member Enumeration Documentation</a>	55
8.4.2.1	<a href="#">EphemerisMode</a>	55

8.4.3	Constructor & Destructor Documentation . . . . .	55
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 . . . . .	56
8.4.5.1	central_point_name . . . . .	56
8.4.5.2	integ_constructor . . . . .	57
8.4.5.3	integ_group_constructor . . . . .	57
8.4.5.4	jeod_integ_opt . . . . .	57
8.4.5.5	mode . . . . .	58
8.4.5.6	sim_integ_opt . . . . .	58
8.5	jeod::DynManagerMessages Class Reference . . . . .	58
8.5.1	Detailed Description . . . . .	59
8.5.2	Constructor & Destructor Documentation . . . . .	59
8.5.2.1	DynManagerMessages() [1/2] . . . . .	59
8.5.2.2	DynManagerMessages() [2/2] . . . . .	59
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 . . . . .	60
8.5.4.2	InputProcessor . . . . .	60
8.5.5	Field Documentation . . . . .	60
8.5.5.1	duplicate_entry . . . . .	60
8.5.5.2	inconsistent_setup . . . . .	61
8.5.5.3	internal_error . . . . .	61
8.5.5.4	invalid_frame . . . . .	61
8.5.5.5	invalid_name . . . . .	61
8.5.5.6	invalid_type . . . . .	62
8.5.5.7	null_pointer . . . . .	62
8.5.5.8	singleton_error . . . . .	62



<b>9 File Documentation</b>	<b>63</b>
9.1 base_dyn_manager.hh File Reference	63
9.1.1 Detailed Description	63
9.2 class_declarations.hh File Reference	63
9.2.1 Detailed Description	64
9.3 dyn_bodies_primitives.cc File Reference	64
9.3.1 Detailed Description	64
9.4 dyn_manager.cc File Reference	64
9.4.1 Detailed Description	65
9.5 dyn_manager.hh File Reference	65
9.5.1 Detailed Description	65
9.6 dyn_manager_init.hh File Reference	66
9.6.1 Detailed Description	66
9.7 dyn_manager_messages.cc File Reference	66
9.7.1 Detailed Description	66
9.7.2 Macro Definition Documentation	67
9.7.2.1 MAKE_DYNMANAGER_MESSAGE_CODE	67
9.8 dyn_manager_messages.hh File Reference	67
9.8.1 Detailed Description	67
9.9 dynamics_integration_group.cc File Reference	67
9.9.1 Detailed Description	68
9.10 dynamics_integration_group.hh File Reference	68
9.10.1 Detailed Description	68
9.11 gravitation.cc File Reference	68
9.11.1 Detailed Description	69
9.12 initialize_dyn_bodies.cc File Reference	69
9.12.1 Detailed Description	69
9.13 initialize_model.cc File Reference	69
9.13.1 Detailed Description	70
9.14 initialize_simulation.cc File Reference	70
9.14.1 Detailed Description	70
9.15 integ_group_primitives.cc File Reference	70
9.15.1 Detailed Description	71
9.16 mass_bodies_primitives.cc File Reference	71
9.16.1 Detailed Description	71
9.17 perform_actions.cc File Reference	71
9.17.1 Detailed Description	71
<b>Index</b>	<b>73</b>



# 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:

<a href="#">jeod</a>	Namespace jeod . . . . .	<a href="#">15</a>
----------------------	--------------------------	--------------------



## 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 . . . . .	54
jeod::DynManagerMessages . . . . .	58
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:

<a href="#">jeod::BaseDynManager</a>	
The <a href="#">DynManager</a> class augments the EphemManager with dynamics-related items	17
<a href="#">jeod::DynamicsIntegrationGroup</a>	
A <a href="#">DynamicsIntegrationGroup</a> integrates the state of a set of DynBoby objects over time	24
<a href="#">jeod::DynManager</a>	
Manages the dynamic elements of a simulation	33
<a href="#">jeod::DynManagerInit</a>	
This class contains data used to initialize a <a href="#">DynManager</a> object	54
<a href="#">jeod::DynManagerMessages</a>	
Specifies the message IDs used in the <a href="#">DynManager</a> model	58



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

<a href="#">base_dyn_manager.hh</a>	Define the BaseDynManager class, which defines the interfaces to the class DynManager . . .	63
<a href="#">class_declarations.hh</a>	Forward declarations of classes defined in <a href="#">dyn_manager.hh</a> . . . . .	63
<a href="#">dyn_bodies_primitives.cc</a>	Define the DynManager member functions that search through and add elements to the collection of DynBody pointers . . . . .	64
<a href="#">dyn_manager.cc</a>	Define simple member functions for the DynManager and related classes . . . . .	64
<a href="#">dyn_manager.hh</a>	Define the DynManager class, which manages the planets and vehicles modeled in a JEOD-based simulation . . . . .	65
<a href="#">dyn_manager_init.hh</a>	Define the DynManagerInit class, which contains the data used to initialize a DynManager object	66
<a href="#">dyn_manager_messages.cc</a>	Implement the class DynManagerMessages . . . . .	66
<a href="#">dyn_manager_messages.hh</a>	Define the class DynManagerMessages, the class that specifies the message IDs used in the DynManager model . . . . .	67
<a href="#">dynamics_integration_group.cc</a>	Define DynamicsIntegrationGroup methods . . . . .	67
<a href="#">dynamics_integration_group.hh</a>	Define the extensible class DynamicsIntegrationGroup, an instance of which is responsible for integrating the states of a set of DynBody objects . . . . .	68
<a href="#">gravitation.cc</a>	Compute gravitational acceleration . . . . .	68
<a href="#">initialize_dyn_bodies.cc</a>	Define DynManager::initialize_dyn_bodies . . . . .	69
<a href="#">initialize_model.cc</a>	Define DynManager::initialize_model . . . . .	69
<a href="#">initialize_simulation.cc</a>	Define DynManager::initialize_simulation, which completes the initialization of the JEOD dynamics manager . . . . .	70
<a href="#">integ_group_primitives.cc</a>	Define the DynManager member functions that search through and add elements to the collection of DynamicsIntegrationGroup pointers . . . . .	70

[mass\\_bodies\\_primitives.cc](#)

Define the DynManager member functions that search through and add elements to the collection of MassBody pointers . . . . . 71

[perform\\_actions.cc](#)

Define DynManager::perform\_actions . . . . . 71

## 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\\_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 Dynamics↔IntegrationGroup 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.*

#### 6.3.1 Detailed Description





## Chapter 7

# Namespace Documentation

### 7.1 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.*

#### 7.1.1 Detailed Description

Namespace jeod.



## Chapter 8

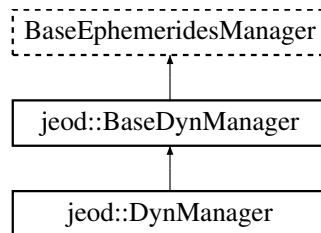
# Data Structure Documentation

### 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

- [~BaseDynManager](#) () override=default
- 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](#) ()=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 std::string &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 std::string &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](#) () 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 81 of file base\_dyn\_manager.hh.

### 8.1.2 Constructor & Destructor Documentation

#### 8.1.2.1 ~BaseDynManager()

```
jeod::BaseDynManager::~~BaseDynManager ( ) [override], [default]
```

### 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 std::string & 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 std::string & 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 ( ) [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 ( ) 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 83 of file `base_dyn_manager.hh`.

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

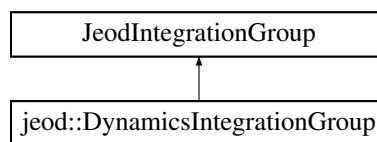
- [base\\_dyn\\_manager.hh](#)

## 8.2 jeod::DynamicsIntegrationGroup Class Reference

A [DynamicsIntegrationGroup](#) integrates the state of a set of DynBoby 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.*
- [~DynamicsIntegrationGroup](#) () override  
*DynamicsIntegrationGroup destructor.*
- [DynamicsIntegrationGroup](#) (const [DynamicsIntegrationGroup](#) &)=delete
- [DynamicsIntegrationGroup](#) & operator= (const [DynamicsIntegrationGroup](#) &)=delete
- bool [is\\_empty](#) () 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](#) ()  
*Collect the forces and torques acting on each root dynamic body.*
- er7\_utils::IntegratorResult [integrate\\_bodies](#) (double cycle\_dyndt, unsigned int target\_stage) override  
*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

- void [reset\\_body\\_integrators](#) () override  
*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](#) {true}  
*This flag is always true for JEOD integration groups.*

## Private Member Functions

- void [register\\_base\\_contents](#) ()  
*Register types and containers.*

## 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 54 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 68 of file `dynamics_integration_group.cc`.

References `register_base_contents()`.

### 8.2.2.3 ~DynamicsIntegrationGroup()

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

[DynamicsIntegrationGroup](#) destructor.

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

References `dyn_bodies`.

### 8.2.2.4 DynamicsIntegrationGroup() [3/3]

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

## 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 166 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 ( ) [virtual]
```

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

Definition at line 298 of file dynamics\_integration\_group.cc.

References dyn\_bodies.

### 8.2.3.3 create\_group()

```
DynamicsIntegrationGroup * jeod::DynamicsIntegrationGroup::create_group (
    JeodIntegrationGroupOwner & owner,
    er7_utils::IntegratorConstructor & integ_cotr,
    JeodIntegratorInterface & integ_inter,
    JeodIntegrationTime & time_mgr ) 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 104 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 222 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 271 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 overriders: 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 135 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 ) [override]
```

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 339 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 ( ) const [inline]
```

Query whether the group is void of registered bodies.

**Returns**

True if group is empty, false otherwise.

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

References `dyn_bodies`.

**8.2.3.9 operator=()**

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

**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 261 of file `dynamics_integration_group.cc`.

**8.2.3.11 register\_base\_contents()**

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

Register types and containers.

Definition at line 80 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 118 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 ( ) [override], [protected]
```

Force all integrators to reset themselves.

Definition at line 319 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 93 of file `dynamics_integration_group.hh`.

## 8.2.5 Field Documentation

### 8.2.5.1 bodies\_integrated\_separately

```
bool jeod::DynamicsIntegrationGroup::bodies_integrated_separately {true} [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 225 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 197 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 215 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:

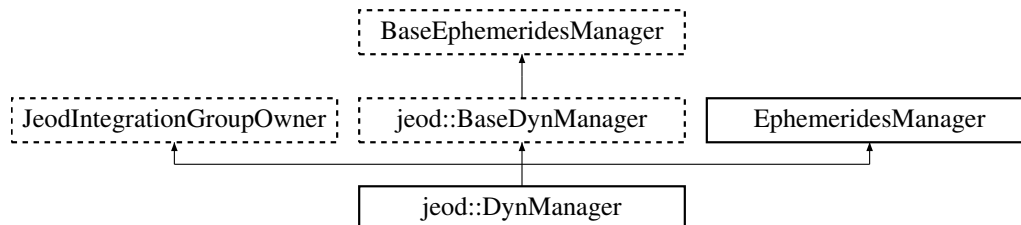
- [dynamics\\_integration\\_group.hh](#)
- [dynamics\\_integration\\_group.cc](#)

## 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.
- [~DynManager](#) () override  
*DynManager* destructor.
- [DynManager](#) (const [DynManager](#) &)=delete
- [DynManager](#) & operator= (const [DynManager](#) &)=delete
- 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](#) ()  
*Complete initialization of the JEOD manager model.*
- void [set\\_gravity\\_manager](#) ([GravityManager](#) &gravity) override  
*Set the Gravity Manager to the specified reference.*
- void [initialize\\_gravity\\_controls](#) () override  
*Initialize the gravity controls for each dynamic body.*
- void [reset\\_gravity\\_controls](#) () override  
*Reset the gravity controls for each dynamic body.*
- void [gravitation](#) ()  
*Compute gravitational acceleration on each root body.*
- void [add\\_mass\\_body](#) ([MassBody](#) &mass\_body) override  
*Add a mass body to the mass body registry.*
- void [add\\_mass\\_body](#) ([MassBody](#) \*mass\_body) override  
*Add a mass body to the mass body registry.*
- [MassBody](#) \* [find\\_mass\\_body](#) (const std::string &name) const override  
*Find the mass body with the given name.*
- bool [is\\_mass\\_body\\_registered](#) (const [MassBody](#) \*mass\_body) const override  
*Determine if the specified body has been registered with the DynManager.*
- void [add\\_dyn\\_body](#) ([DynBody](#) &dyn\_body) override

- Add a dynamic body to the dynamic body registry.*

  - DynBody \* [find\\_dyn\\_body](#) (const std::string &name) const override

*Find the dynamic body with the given name.*
- std::vector< DynBody \* > [get\\_dyn\\_bodies](#) () const override

*Return a copy of the list of registered dynamic bodies.*
- bool [is\\_dyn\\_body\\_registered](#) (const DynBody \*dyn\_body) const override

*Determine if the specified body has been registered with the [DynManager](#).*
- void [add\\_integ\\_group](#) (DynamicsIntegrationGroup &integ\_group) override

*Add an integration group to the integration group registry.*
- bool [is\\_integ\\_group\\_registered](#) (const DynamicsIntegrationGroup \*integ\_group) const override

*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](#) (const std::string &action\_name\_in)

*Remove a body action to the list of such.*
- void [perform\\_actions](#) ()

*Perform dynamic body actions that are ready to be applied.*
- void [initialize\\_integ\\_groups](#) ()

*Complete initialization of the initialization groups.*
- void [update\\_integration\\_group](#) (JeodIntegrationGroup &group) override

*Add DynBody objects to the default integration group.*
- void [initialize\\_dyn\\_bodies](#) ()

*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.*
- void [reset\\_integrators](#) () override

*Force all integrators to reset themselves.*
- void [reset\\_integrators](#) (DynamicsIntegrationGroup &integ\_group) override

*Instruct specific integrator to reset itself.*
- int [integrate](#) (double to\_sim\_time, TimeManager &)

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

*Return last update time.*
- const std::string [name](#) () const

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

*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 {DynManagerInit::EphemerisMode\_Ephemerides}

*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=nullptr)  
*Initialize all queued body actions that derive from MassBodyInit and apply those that are immediately ready to be applied.*
- void [perform\\_mass\\_attach\\_initializations](#) ()  
*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=nullptr)  
*Initialize dynamic bodies.*
- void [check\\_for\\_uninitialized\\_states](#) ()  
*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.*

## 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 113 of file dyn\_manager.hh.

### 8.3.2 Constructor & Destructor Documentation

#### 8.3.2.1 DynManager() [1/2]

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

[DynManager](#) default constructor.

Definition at line 65 of file dyn\_manager.cc.

#### 8.3.2.2 ~DynManager()

```
jeod::DynManager::~~DynManager ( ) [override]
```

[DynManager](#) destructor.

Definition at line 84 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 & ) [delete]
```

## 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 168 of file `dyn_manager.cc`.

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

### 8.3.3.2 add\_dyn\_body()

```
void jeod::DynManager::add_dyn_body (
    DynBody & dyn_body ) [override], [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 95 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 ) [override], [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 62 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 ) [override], [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 89 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 ) [override], [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 128 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 ( ) [protected]
```

Ensure that all of the required states have been set.

Definition at line 337 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 216 of file `dyn_manager.hh`.

#### 8.3.3.8 `find_dyn_body()`

```
DynBody * jeod::DynManager::find_dyn_body (
    const std::string & body_name ) const [override], [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 55 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 std::string & body_name ) const [override], [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 49 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()**

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

Return a copy of the list of registered dynamic bodies.

**Returns**

Copy of dyn\_bodies data member

Implements [jeod::BaseDynManager](#).

Definition at line 178 of file dyn\_manager.hh.

**8.3.3.11 gravitation()**

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

Compute gravitational acceleration on each root body.

Definition at line 119 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 ( )
```

Initialize dynamic bodies.

Definition at line 55 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 99 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 ( ) [override], [virtual]
```

Initialize the gravity controls for each dynamic body.

##### Assumptions and Limitations

- Not called in empty space mode.

Implements [jeod::BaseDynManager](#).

Definition at line 49 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 ( )
```

Complete initialization of the initialization groups.

Definition at line 100 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 61 of file initialize\_model.cc.

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

Class: (initialization)

Definition at line 78 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</i> , out	<i>init</i>	Initialization data
<i>in</i> , out	<i>time_mngr</i>	Time manager

Definition at line 95 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 ( )
```

Complete initialization of the JEOD manager model.

Definition at line 46 of file initialize\_simulation.cc.

References [jeod::DynManagerInit::EphemerisMode\\_EmptySpace](#), [gravity\\_manager](#), [gravity\\_off](#), [jeod::DynManagerMessages::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

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

#### Returns

zero if complete, non-zero if incomplete.

Definition at line 240 of file dyn\_manager.hh.

### 8.3.3.21 is\_dyn\_body\_registered()

```
bool jeod::DynManager::is_dyn_body_registered (
    const DynBody * dyn_body ) const [override], [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 86 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 131 of file dyn\_manager.hh.

#### 8.3.3.23 `is_integ_group_registered()`

```
bool jeod::DynManager::is_integ_group_registered (
    const DynamicsIntegrationGroup * integ_group ) const [override], [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 53 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 [override], [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 80 of file `mass_bodies_primitives.cc`.

References `mass_bodies`.

Referenced by `add_mass_body()`.

**8.3.3.25 name()**

```
const std::string jeod::DynManager::name ( ) const
```

Return identifier.

**Returns**

Name

Definition at line 108 of file `dyn_manager.cc`.

**8.3.3.26 operator=()**

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

**8.3.3.27 perform\_actions()**

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

Perform dynamic body actions that are ready to be applied.

Definition at line 41 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 = nullptr ) [protected]
```

Initialize dynamic bodies.



## Parameters

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

Definition at line 220 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 ( ) [protected]
```

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

Definition at line 171 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 = nullptr ) [protected]
```

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

## Parameters

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

Definition at line 116 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 (
    const std::string & action_name_in )
```

Remove a body action to the list of such.

**Parameters**

in	<i>action_name</i> ↔ <i>_in</i>	Name of the action to remove
----	------------------------------------	------------------------------

Definition at line 205 of file dyn\_manager.cc.

References body\_actions.

**8.3.3.32 reset\_gravity\_controls()**

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

Reset the gravity controls for each dynamic body.

**Assumptions and Limitations**

- Not called in empty space mode.

Implements [jeod::BaseDynManager](#).

Definition at line 82 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 ( ) [override], [virtual]
```

Force all integrators to reset themselves.

Implements [jeod::BaseDynManager](#).

Definition at line 227 of file dyn\_manager.cc.

References default\_integ\_group, and integ\_groups.

**8.3.3.34 reset\_integrators()** [2/2]

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

Instruct specific integrator to reset itself.

## Parameters

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

Implements [jeod::BaseDynManager](#).

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

**8.3.3.35 set\_gravity\_manager()**

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

Set the Gravity Manager to the specified reference.

## Parameters

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

Implements [jeod::BaseDynManager](#).

Definition at line 125 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 ( )
```

Shutdown the manager.

Empty for now.

Definition at line 116 of file `dyn_manager.cc`.

**8.3.3.37 timestamp()**

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

Return last update time.

## Returns

Name

Implements [jeod::BaseDynManager](#).

Definition at line 99 of file `dyn_manager.cc`.

### 8.3.3.38 update\_integration\_group()

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

Add DynBody objects to the default integration group.

#### Parameters

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

Definition at line 126 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 117 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 340 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 315 of file dyn\_manager.hh.

Referenced by add\_integ\_group(), gravitation(), initialize\_integ\_groups(), initialize\_model\_internal(), 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 259 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 330 of file dyn\_manager.hh.

Referenced by add\_dyn\_body(), check\_for\_uninitialized\_states(), find\_dyn\_body(), 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 300 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 266 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 295 of file `dyn_manager.hh`.

Referenced by `add_body_action()`, `add_integ_group()`, `gravitation()`, `initialize_simulation()`, 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 305 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 335 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 310 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 325 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 {DynManagerInit::EphemerisMode_Ephemerides}
```

The ephemeris mode in which the dynamics manager operates.

trick\_units(–)

Definition at line 271 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 276 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 320 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](#) ()=default
- [~DynManagerInit](#) ()=default
- [DynManagerInit](#) (const [DynManagerInit](#) &)=delete
- [DynManagerInit](#) & operator= (const [DynManagerInit](#) &)=delete



## Data Fields

- [EphemerisMode](#) `mode` {[EphemerisMode\\_Ephemerides](#)}  
*Dynamics manager mode.*
- `std::string` [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](#) {`er7_utils::Integration::Unspecified`}  
*Integrator type.*
- `int` [sim\\_integ\\_opt](#) {-1}  
*Integrator type.*

### 8.4.1 Detailed Description

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

Definition at line 80 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

<code>EphemerisMode_EmptySpace</code>	
<code>EphemerisMode_SinglePlanet</code>	
<code>EphemerisMode_Ephemerides</code>	

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

### 8.4.3 Constructor & Destructor Documentation

#### 8.4.3.1 DynManagerInit() [1/2]

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

#### 8.4.3.2 ~DynManagerInit()

```
jeod::DynManagerInit::~~DynManagerInit ( ) [default]
```

#### 8.4.3.3 DynManagerInit() [2/2]

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

### 8.4.4 Member Function Documentation

#### 8.4.4.1 operator=()

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

### 8.4.5 Field Documentation

#### 8.4.5.1 central\_point\_name

```
std::string 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 116 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 150 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 129 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 {er7_utils::Integration↵
::Unspecified}
```

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 157 of file dyn\_manager\_init.hh.

Referenced by jeod::DynManager::initialize\_model\_internal().

#### 8.4.5.5 mode

```
EphemerisMode jeod::DynManagerInit::mode {EphemerisMode_Ephemerides}
```

Dynamics manager mode.

trick\_units(-)

Definition at line 110 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 {-1}
```

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 164 of file dyn\_manager\_init.hh.

Referenced by jeod::DynManager::initialize\_model\_internal().

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

- [dyn\\_manager\\_init.hh](#)

## 8.5 jeod::DynManagerMessages Class Reference

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

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

### Public Member Functions

- [DynManagerMessages](#) ()=delete
- [DynManagerMessages](#) (const [DynManagerMessages](#) &)=delete
- [DynManagerMessages](#) & operator= (const [DynManagerMessages](#) &)=delete

## Static Public Attributes

- static const char \* [null\\_pointer](#) = "dynamics/dyn\_manager/" "null\_pointer"  
*Issued when a pointer should be non-NULL but isn't.*
- static const char \* [duplicate\\_entry](#) = "dynamics/dyn\_manager/" "duplicate\_entry"  
*Issued on request to add a pointer to a list a second time.*
- static const char \* [invalid\\_name](#) = "dynamics/dyn\_manager/" "invalid\_name"  
*Issued when a name is invalid – empty, a duplicate, ...*
- static const char \* [invalid\\_frame](#) = "dynamics/dyn\_manager/" "invalid\_frame"  
*Issued when a frame is invalid – not an integ frame, ...*
- static const char \* [invalid\\_type](#) = "dynamics/dyn\_manager/" "invalid\_type"  
*Issued when an object of an unexpected type is encountered.*
- static const char \* [inconsistent\\_setup](#) = "dynamics/dyn\_manager/" "inconsistent\_setup"  
*Issued when some conditions are inconsistent.*
- static const char \* [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 const char \* [internal\\_error](#) = "dynamics/dyn\_manager/" "internal\_error"  
*Error issued when some internal error occurred.*

## Friends

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

### 8.5.1 Detailed Description

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

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

### 8.5.2 Constructor & Destructor Documentation

#### 8.5.2.1 DynManagerMessages() [1/2]

```
jeod::DynManagerMessages::DynManagerMessages ( ) [delete]
```

#### 8.5.2.2 DynManagerMessages() [2/2]

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

### 8.5.3 Member Function Documentation

#### 8.5.3.1 operator=()

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

### 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 83 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 93 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 113 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 125 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 103 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 98 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 108 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 88 of file dyn\_manager\_messages.hh.

Referenced by 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 119 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/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.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.hh File Reference

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

```
#include <string>
#include "er7_utils/integration/core/include/integration_technique.hh"
#include "er7_utils/integration/core/include/integrator_constructor.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

- [jeod](#)

*Namespace jeod.*

#### 9.6.1 Detailed Description

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

## 9.7 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(DynManager↵ Messages, "dynamics/dyn\_manager/", id)

#### 9.7.1 Detailed Description

Implement the class DynManagerMessages.

## 9.7.2 Macro Definition Documentation

### 9.7.2.1 MAKE\_DYNMANAGER\_MESSAGE\_CODE

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

Definition at line 37 of file dyn\_manager\_messages.cc.

## 9.8 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.*

### 9.8.1 Detailed Description

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

## 9.9 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/memory/include/jeod_alloc.hh"  
#include "utils/message/include/message_handler.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.*

### 9.9.1 Detailed Description

Define DynamicsIntegrationGroup methods.

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

## Data Structures

- class [jeod::DynamicsIntegrationGroup](#)

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

## Namespaces

- [jeod](#)

*Namespace jeod.*

### 9.10.1 Detailed Description

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

## 9.11 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.*

### 9.11.1 Detailed Description

Compute gravitational acceleration.

## 9.12 initialize\_dyn\_bodies.cc File Reference

Define DynManager::initialize\_dyn\_bodies.

```
#include <cstdint>
#include "dynamics/body_action/include/body_action.hh"
#include "dynamics/body_action/include/body_attach.hh"
#include "dynamics/body_action/include/dyn_body_init.hh"
#include "dynamics/body_action/include/mass_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.*

### 9.12.1 Detailed Description

Define DynManager::initialize\_dyn\_bodies.

## 9.13 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.*

### 9.13.1 Detailed Description

Define DynManager::initialize\_model.

## 9.14 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.*

### 9.14.1 Detailed Description

Define DynManager::initialize\_simulation, which completes the initialization of the JEOD dynamics manager.

## 9.15 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 <cstdint>
#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 the DynManager member functions that search through and add elements to the collection of Dynamics↔ IntegrationGroup pointers.

## 9.16 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 <cstddef>
#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.*

### 9.16.1 Detailed Description

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

## 9.17 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.*

### 9.17.1 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, [63](#)
- bodies\_integrated\_separately
  - jeod::DynamicsIntegrationGroup, [31](#)
- body\_actions
  - jeod::DynManager, [50](#)
- central\_point\_name
  - jeod::DynManagerInit, [56](#)
- check\_for\_uninitialized\_states
  - jeod::DynManager, [39](#)
- class\_declarations.hh, [63](#)
- collect\_derivatives
  - jeod::DynamicsIntegrationGroup, [27](#)
- compute\_derivatives
  - jeod::DynManager, [39](#)
- create\_group
  - jeod::DynamicsIntegrationGroup, [27](#)
- default\_integ\_group
  - jeod::DynManager, [50](#)
- delete\_dyn\_body
  - jeod::DynamicsIntegrationGroup, [28](#)
- deriv\_ephem\_update
  - jeod::DynManager, [51](#)
  - jeod::DynamicsIntegrationGroup, [32](#)
- duplicate\_entry
  - jeod::DynManagerMessages, [60](#)
- dyn\_bodies
  - jeod::DynManager, [51](#)
  - jeod::DynamicsIntegrationGroup, [32](#)
- dyn\_bodies\_primitives.cc, [64](#)
- dyn\_manager.cc, [64](#)
- dyn\_manager.hh, [65](#)
- dyn\_manager\_init.hh, [66](#)
- dyn\_manager\_messages.cc, [66](#)
- MAKE\_DYNMANAGER\_MESSAGE\_CODE, [67](#)
- dyn\_manager\_messages.hh, [67](#)
- DynManager, [13](#)
  - jeod::DynManager, [36](#)
- DynManagerInit
  - jeod::DynManagerInit, [55, 56](#)
- DynManagerMessages
  - jeod::DynManagerMessages, [59](#)
- Dynamics, [12](#)
- dynamics\_integration\_group.cc, [67](#)
- dynamics\_integration\_group.hh, [68](#)
- DynamicsIntegrationGroup
  - jeod::DynamicsIntegrationGroup, [26](#)
- EphemerisMode
  - jeod::DynManagerInit, [55](#)
- 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, [68](#)
- gravity\_manager
  - jeod::DynManager, [51](#)
- gravity\_off
  - jeod::DynManager, [51](#)
- inconsistent\_setup
  - jeod::DynManagerMessages, [60](#)
- 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, 69
- initialize\_dyn\_body
  - jeod::DynManager, 41
- initialize\_gravity\_controls
  - jeod::BaseDynManager, 21
  - jeod::DynManager, 41
- initialize\_group
  - jeod::DynamicsIntegrationGroup, 29
- initialize\_integ\_groups
  - jeod::DynManager, 42
- initialize\_model
  - jeod::DynManager, 42
- initialize\_model.cc, 69
- initialize\_model\_internal
  - jeod::DynManager, 43
- initialize\_simulation
  - jeod::DynManager, 43
- initialize\_simulation.cc, 70
- initialized
  - jeod::DynManager, 52
- InputProcessor
  - jeod::BaseDynManager, 23
  - jeod::DynManager, 50
  - jeod::DynManagerMessages, 60
  - jeod::DynamicsIntegrationGroup, 31
- integ\_constructor
  - jeod::DynManager, 52
  - jeod::DynManagerInit, 56
- integ\_group\_constructor
  - jeod::DynManagerInit, 57
- integ\_group\_primitives.cc, 70
- integ\_groups
  - jeod::DynManager, 52
- integ\_interface
  - jeod::DynManager, 52
- integrate
  - jeod::DynManager, 44
- integrate\_bodies
  - jeod::DynamicsIntegrationGroup, 29
- internal\_error
  - jeod::DynManagerMessages, 61
- invalid\_frame
  - jeod::DynManagerMessages, 61
- invalid\_name
  - jeod::DynManagerMessages, 61
- invalid\_type
  - jeod::DynManagerMessages, 61
- 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, 45
- 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, 50
  - check\_for\_uninitialized\_states, 39
  - compute\_derivatives, 39
  - default\_integ\_group, 50
  - deriv\_ephem\_update, 51
  - dyn\_bodies, 51
  - DynManager, 36
  - find\_dyn\_body, 39
  - find\_mass\_body, 40
  - get\_dyn\_bodies, 40
  - gravitation, 40
  - gravity\_manager, 51
  - gravity\_off, 51
  - init\_attrjeod\_\_DynManager, 50
  - initialize\_dyn\_bodies, 41
  - initialize\_dyn\_body, 41
  - initialize\_gravity\_controls, 41
  - initialize\_integ\_groups, 42
  - initialize\_model, 42
  - initialize\_model\_internal, 43
  - initialize\_simulation, 43
  - initialized, 52
  - InputProcessor, 50
  - integ\_constructor, 52
  - integ\_groups, 52
  - integ\_interface, 52

- integrate, 44
- is\_dyn\_body\_registered, 44
- is\_initialized, 45
- is\_integ\_group\_registered, 45
- is\_mass\_body\_registered, 45
- mass\_bodies, 53
- mode, 53
- name, 46
- operator=, 46
- perform\_actions, 46
- perform\_dyn\_body\_initializations, 46
- perform\_mass\_attach\_initializations, 47
- perform\_mass\_body\_initializations, 47
- remove\_body\_action, 47
- reset\_gravity\_controls, 48
- reset\_integrators, 48
- set\_gravity\_manager, 49
- shutdown, 49
- sim\_integrator, 53
- simple\_ephemeris, 53
- timestep, 49
- update\_integration\_group, 49
- jeod::DynManagerInit, 54
  - ~DynManagerInit, 56
  - central\_point\_name, 56
  - DynManagerInit, 55, 56
  - EphemerisMode, 55
  - integ\_constructor, 56
  - integ\_group\_constructor, 57
  - jeod\_integ\_opt, 57
  - mode, 57
  - operator=, 56
  - sim\_integ\_opt, 58
- jeod::DynManagerMessages, 58
  - duplicate\_entry, 60
  - DynManagerMessages, 59
  - inconsistent\_setup, 60
  - init\_attrjeod\_\_DynManagerMessages, 60
  - InputProcessor, 60
  - internal\_error, 61
  - invalid\_frame, 61
  - invalid\_name, 61
  - invalid\_type, 61
  - null\_pointer, 62
  - operator=, 60
  - singleton\_error, 62
- jeod::DynamicsIntegrationGroup, 24
  - ~DynamicsIntegrationGroup, 26
  - add\_dyn\_body, 27
  - bodies\_integrated\_separately, 31
  - collect\_derivatives, 27
  - create\_group, 27
  - delete\_dyn\_body, 28
  - deriv\_ephem\_update, 32
  - dyn\_bodies, 32
  - DynamicsIntegrationGroup, 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, 57
- MAKE\_DYNMANAGER\_MESSAGE\_CODE
  - dyn\_manager\_messages.cc, 67
- mass\_bodies
  - jeod::DynManager, 53
- mass\_bodies\_primitives.cc, 71
- mode
  - jeod::DynManager, 53
  - jeod::DynManagerInit, 57
- 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, 46
- perform\_actions.cc, 71
- perform\_dyn\_body\_initializations
  - jeod::DynManager, 46
- 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, 47
- reset\_body\_integrators
  - jeod::DynamicsIntegrationGroup, 31
- reset\_gravity\_controls
  - jeod::BaseDynManager, 22
  - jeod::DynManager, 48
- reset\_integrators
  - jeod::BaseDynManager, 22
  - jeod::DynManager, 48

- set\_gravity\_manager
  - jeod::BaseDynManager, [23](#)
  - jeod::DynManager, [49](#)
- shutdown
  - jeod::DynManager, [49](#)
- sim\_integ\_opt
  - jeod::DynManagerInit, [58](#)
- sim\_integrator
  - jeod::DynManager, [53](#)
- simple\_ephemeris
  - jeod::DynManager, [53](#)
- singleton\_error
  - jeod::DynManagerMessages, [62](#)
- timestamp
  - jeod::BaseDynManager, [23](#)
  - jeod::DynManager, [49](#)
- update\_integration\_group
  - jeod::DynManager, [49](#)