MassTreeModel

5.3

Generated by Doxygen 1.8.14

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Module Documentation

6.1 Models

Modules

- Dynamics

6.1.1 Detailed Description

6.2 Dynamics

Modules

- Mass

6.2.1 Detailed Description

6.3 Mass

Files

- file class_declarations.hh

  Forward declarations of mass model classes.
- file mass.hh

  Define the class MassBody.
- file mass_body_links.hh

  Define the class MassBodyLinks, the class that encapsulates the links between mass bodies.
- file mass_messages.hh

  Define the class MassBodyMessages, the class that specifies the message IDs used in the MassBody model.
- file mass_point.hh

  Define the class MassPoint, which defines the base features of a point related to a MassBody.
- file mass_point_init.hh

  Define the class MassPointInit, which initializes a MassPoint object.
- file mass_point_links.hh

  Define the class MassPointLinks, the class that encapsulates the links between mass points.
- file mass_point_state.hh

  Define the class MassPointState, which defines the state – position and orientation – of a MassPoint.
- file mass_properties.hh

  Define the class MassProperties.
- file mass_properties_init.hh

  Define class MassPropertiesInit, which initializes a mass properties object.
- file mass.cc

  Define methods for the MassBody class.
- file mass_attach.cc

  Define MassBody attach methods.
- file mass_calc_composite_cm.cc

  Calculate the aggregate cm for a composite MassBody.
- file mass_calc_composite_inertia.cc

  Calculate the aggregate inertia matrix for a composite MassBody.
- file mass_detach.cc

  Define MassBody detachment methods.
- file mass_messages.cc

  Implement the class MassBodyMessages.
- file mass_point.cc

  Define basic methods for the MassPoint class.
- file mass_point_init.cc

  Define methods for the MassPointInit class.
- file mass_point_mass_inertia.cc

  Compute the inertia tensor of a point mass.
- file mass_point_state.cc

  Define basic methods for the MassPointState class.
- file mass_print_body.cc

  Define MassBody::print_body.
- file mass_print_tree.cc

  Print out the MassTree members' data to a given file.
- file mass_properties_init.cc

  Define methods for the MassPropertiesInit class.
- file mass_reattach.cc

  Define MassBody::reattach.
- file mass_update.cc

  Define MassBody::update_mass_properties.

Namespaces

- jeod

    Namespace jeod.

6.3.1 Detailed Description

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class MassBody

  Represent both an atomic chunk of mass and an interconnected set of masses.
- class MassBodyLinks

  Encapsulates the links between mass bodies.
- class MassBodyMessages

  Specify the message IDs used in the MassBody model.
- class MassPoint

  Adds tree linkages and a name to a MassPointState.
- class MassPointInit

  Contains data used to initialize a MassPoint.
- class MassPointLinks

  Encapsulates the links between mass points.
- class MassPointState

  Defines the state – position and orientation – of a MassPoint.
- class MassProperties

  Defines mass properties – mass and inertia tensor.
- class MassPropertiesInit

  Contains data used to initialize a mass model object.

Functions

- static void generate_bad_point_message (const char ∗file, unsigned int line, const std::string &child_body_name, const std::string &child_point_name, const MassPoint ∗child_point, const std::string &parent_body_name, const std::string &parent_point_name, const MassPoint ∗parent_point)

  Generate a message regarding failure to find mass points on a vehicle.

7.1.1 Detailed Description

Namespace jeod.

7.1.2 Function Documentation

7.1.2.1 generate_bad_point_message()

```
static void jeod::generate_bad_point_message (
             const char * file,
             unsigned int line,
             const std::string & child_body_name,
             const std::string & child_point_name,
             const MassPoint * child_point,
             const std::string & parent_body_name,
             const std::string & parent_point_name,
             const MassPoint * parent_point )  [static]
```

Generate a message regarding failure to find mass points on a vehicle.

Assumptions and Limitations

- One or both of the input MassPoint pointers is null.

Parameters

in	*file*	File name
in	*line*	Line number
in	*child_body_name*	Name of child body
in	*child_point_name*	Name of child mass point
in	*child_point*	Child mass point
in	*parent_body_name*	Name of parent body
in	*parent_point_name*	Name of parent mass point
in	*parent_point*	Parent mass point

Definition at line 543 of file mass_attach.cc.

References jeod::MassBodyMessages::invalid_attach.

Referenced by jeod::MassBody::attach_to().

Chapter 8

Data Structure Documentation

8.1 jeod::MassBody Class Reference

Represent both an atomic chunk of mass and an interconnected set of masses.

```
#include <mass.hh>
```

Public Member Functions

- MassBody ()

  Default constructor; constructs a MassBody object.
- virtual ∼MassBody ()

  Destroy a MassBody object.
- MassBody (const MassBody &)=delete
- void operator= (const MassBody &)=delete
- void initialize_mass (const MassPropertiesInit &properties, const std::vector< MassPointInit *> &points)

  Initialize a MassBody object.
- bool get_mass_properties_initialized ()
- void set_name (std::string name_in)
- virtual bool attach_to (const std::string &this_point_name, const std::string &parent_point_name, MassBody &parent)

  Attach this mass body's root body as a child of the specified mass body such that the specified mass points on the two bodies are coincident and the frames associated with those mass points are related by a 180 degree yaw.
- virtual bool attach_to (double offset_pstr_cstr_pstr[3], double T_pstr_cstr[3][3], MassBody &parent)

  Attach this mass body's root body as a child of the specified mass body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with respect to the parent body's structural axes as specified.
- virtual bool attach_child (const std::string &this_point_name, const std::string &child_point_name, MassBody &child)

  Attach a child MassBody by point specification.
- virtual bool attach_child (double offset_pstr_cstr_pstr[3], double T_pstr_cstr[3][3], MassBody &child)

  Attach a child MassBody by location specification.
- virtual bool attach_validate (const MassBody &parent, bool generate_message) const

  Validate whether the pending attachment is legal.
- virtual bool detach (MassBody &from_body)

Detach the two bodies, 'this' and the argument body, from each other such such that the detachment occurs between the superior body and the immediate child along the path from the inferior body to the superior body.

- virtual bool detach ()

  Detach a mass body from its immediate parent.
- virtual bool detach_validate (const MassBody ∗parent, bool generate_message) const

  Validate whether the pending detachment is legal.
- virtual bool reattach (double offset[3], double T_pstr_cstr[3][3])

  Re-attach a child MassBody to a parent MassBody.
- const MassPoint ∗ find_mass_point (const std::string &pt_name) const

  Find the mass point with the given name.
- virtual void add_mass_point (const MassPointInit &mass_point_init)

  Add a mass point to the list of such.
- std::size_t mass_points_size () const

  Return the number of mass points for this body.
- bool is_progeny_of (const MassBody &test_body) const

  Return true if this MassBody is an offspring of provided one, false if not.
- virtual const MassBody ∗ get_parent_body () const

  Returns the MassBody's parent body, as a const pointer.
- virtual const MassBody ∗ get_root_body () const

  Finds & returns root of current MassBody's tree.
- void set_update_flag ()

  Flag mass bodies from the current body on up the mass tree as in need of mass property updates.
- virtual void update_mass_properties ()

  Update composite mass properties for a mass tree marked for update.
- void print_body (FILE ∗file_ptr, int levels) const

  Recursively print out the data in this and its children to given file.
- void print_tree (const char ∗file_name, int levels) const

  Print out the MassTree members' data to a given file.

Static Public Member Functions

- static void compute_point_mass_inertia (double mass, const double r_pt[3], double inertia[3][3])

  Compute the inertia tensor of a point mass.

Data Fields

- NamedItem name

  Body name.
- MassProperties core_properties

  The mass properties of this body alone, excluding child bodies.
- MassProperties composite_properties

  The mass properties of this body, including child bodies.
- bool compute_inverse_inertia {}

  When clear, the inverse of the inertia tensor is not calculated.
- MassPoint structure_point

  The location and orientation of the structural frame with respect to the parent body's structural frame.
- DynBody ∗const dyn_owner {}

  Indicates this body belongs to a DynBody.

Protected Member Functions

- MassBody (DynBody &owner)

  Create a MassBody, marking is_dynamic as true.
- virtual bool attach_root_body (double offset_pstr_cstr_pstr[3], double T_pstr_cstr[3][3], MassBody &parent)

  Attach this mass body, which must be a root body, as a child of the specified mass body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with repect to the parent body's structural axes as specified.
- virtual bool attach_validate_parent (const MassBody &parent, bool generate_message) const

  Validate whether the pending attachment is legal from a connectivity point of view.
- virtual bool attach_validate_child (const MassBody &child, bool generate_message) const

  Validate whether the pending attachment is legal from a physical point of view.
- virtual void attach_establish_links (MassBody &parent)

  Establish the logical connectivity between parent and child.
- virtual void attach_update_properties (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3], MassBody &child)

  Set the relation between parent and child and update the mass properties.
- virtual bool detach_validate_parent (const MassBody ∗parent, bool generate_message) const

  Validate whether the pending detachment is legal from a connectivity point of view.
- virtual bool detach_validate_child (const MassBody &child, bool generate_message) const

  Validate whether the pending detachment is legal from a mass tree point of view.
- virtual void detach_sever_links (MassBody &parent)

  Break the logical connectivity between parent and child.
- virtual void detach_update_properties (MassBody &child)

  Update parent and child properties to reflect that they are detached.
- virtual MassBody ∗ get_parent_body_internal ()

  Returns the MassBody's parent body, as a non-const pointer.
- virtual MassBody ∗ get_root_body_internal ()

  Finds & returns root of current MassBody's tree.
- void calc_composite_cm ()

  Calculate the aggregate cm for a composite MassBody.
- void calc_composite_inertia ()

  Calculate the aggregate inertia matrix for a composite MassBody.

Protected Attributes

- BaseDynManager ∗ dyn_manager {}

  The dynamics manager for the simulation.
- bool mass_properties_initialized {}

  Initialized flag.
- MassBodyLinks links

  Linkage to rest of mass tree.
- bool needs_update {}

  When set, indicates that an update to the composite mass is needed.
- std::list< MassPoint ∗ > mass_points

  List of points associated with this mass body.
- MassPoint core_wrt_composite

  The location and orientation of the composite CoM and body frame with respect to the parent body's composite CoM and body frame.
- MassPoint composite_wrt_pstr

  The location and orientation of the composite CoM and body frame with respect to the parent body's structural frame.
- MassPoint composite_wrt_pbdy

  The location and orientation of the composite CoM and body frame with respect to the parent body's composite CoM and body frame.

Friends

- class InputProcessor
- class MassBodyLinks
- class DynBody
- void init_attrjeod__MassBody ()

8.1.1 Detailed Description

Represent both an atomic chunk of mass and an interconnected set of masses.

Multiple MassBody objects can be attached to one another in a tree structure. This connected sets of bodies has a composite set of mass properties. Sans these connections, a body will have some core mass proprties.

The MassBody class is one of the key classes in JEOD 2.0. In addition to representing masses, it is also the parent of the DynBody class, which is used in JEOD 2.0 to represent vehicles.

Assumptions and Limitations

- Rigid Bodies
- Tree attachment Structure

Definition at line 112 of file mass.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 MassBody() [1/3]

```
jeod::MassBody::MassBody ( )
```

Default constructor; constructs a MassBody object.

Definition at line 76 of file mass.cc.

8.1.2.2 ∼MassBody()

```
jeod::MassBody::∼MassBody ( )  [virtual]
```

Destroy a MassBody object.

Definition at line 140 of file mass.cc.

References detach(), jeod::MassPoint::detach(), jeod::MassBodyMessages::invalid_detach, links, mass_points, name, and set_update_flag().

8.1.2.3 MassBody() [2/3]

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

8.1.2.4 MassBody() [3/3]

```
jeod::MassBody::MassBody (
            DynBody & owner )  [explicit], [protected]
```

Create a MassBody, marking is_dynamic as true.

Default constructor; constructs a MassBody object.

Utilized by DynBody via friendship.

Definition at line 107 of file mass.cc.

References jeod::MassPoint::attach(), composite_properties, composite_wrt_pbdy, core_properties, core_wrt_↩ composite, structure_point, and jeod::MassPointState::T_parent_this.

8.1.3 Member Function Documentation

8.1.3.1 add_mass_point()

```
void jeod::MassBody::add_mass_point (
            const MassPointInit & mass_point_init )  [virtual]
```

Add a mass point to the list of such.

Parameters

in	*mass_point_init*	Mass point spec

Definition at line 339 of file mass.cc.

References jeod::MassPoint::attach(), find_mass_point(), jeod::MassPointInit::initialize_mass_point(), jeod::↩ MassBodyMessages::invalid_name, mass_points, jeod::MassPoint::name, jeod::MassPointInit::name, name, and structure_point.

Referenced by initialize_mass().

8.1.3.2 attach_child() [1/2]

```
bool jeod::MassBody::attach_child (
            const std::string & this_point_name,
            const std::string & child_point_name,
            MassBody & child )  [virtual]
```

Attach a child MassBody by point specification.

See corresponding MassBody::attach_to() method for more information.

Definition at line 231 of file mass_attach.cc.

References attach_to().

8.1.3.3 attach_child() [2/2]

```
bool jeod::MassBody::attach_child (
            double offset_pstr_cstr_pstr[3],
            double T_pstr_cstr[3][3],
            MassBody & child )  [virtual]
```

Attach a child MassBody by location specification.

See corresponding MassBody::attach_to() method for more information. Note that the offset and transformation are specified w.r.t. the parent in both attach_to() and attach_child()

Definition at line 244 of file mass_attach.cc.

References attach_to().

8.1.3.4 attach_establish_links()

```
void jeod::MassBody::attach_establish_links (
            MassBody & parent )  [protected], [virtual]
```

Establish the logical connectivity between parent and child.

Extensibility comments –

- This method is invoked before the computing the physical relation between parent and child.
- The generic purpose of this method is to establish the logical connectivity between parent and child in terms of the child class.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The attachment is valid; not checked.

Parameters

`in,out`	*parent*	The new parent body; the body to which this body is to be attached.

Definition at line 461 of file mass_attach.cc.

References jeod::MassPoint::attach(), composite_wrt_pbdy, composite_wrt_pstr, links, and structure_point.

Referenced by attach_root_body().

8.1.3.5 attach_root_body()

```
bool jeod::MassBody::attach_root_body (
              double offset_pstr_cstr_pstr[3],
              double T_pstr_cstr[3][3],
              MassBody & parent )  [protected], [virtual]
```

Attach this mass body, which must be a root body, as a child of the specified mass body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with repect to the parent body's structural axes as specified.

Assumptions and Limitations

- The subject body, this, must be a root body.

Returns

Success indicator: true=success, false=attachment not performed.

Parameters

`in`	*offset_pstr_cstr_pstr*	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the new parent body. Units: M
`in`	*T_pstr_cstr*	Transformation matrix from the new parent body's structural frame to this body's structural frame.
`in,out`	*parent*	The new parent body; the body to which this body is to be attached.

Definition at line 267 of file mass_attach.cc.

References attach_establish_links(), attach_update_properties(), attach_validate(), jeod::MassBodyMessages::invalid_attach, links, and name.

Referenced by attach_to().

8.1.3.6 attach_to() [1/2]

```
bool jeod::MassBody::attach_to (
            const std::string & this_point_name,
            const std::string & parent_point_name,
            MassBody & parent )  [virtual]
```

Attach this mass body's root body as a child of the specified mass body such that the specified mass points on the two bodies are coincident and the frames associated with those mass points are related by a 180 degree yaw.

Returns

Success indicator: true=success, false=attachment not performed.

Parameters

in	*this_point_name*	The name of a mass point contained in this mass body's list of mass points.
in	*parent_point_name*	The name of a mass point contained in the new parent body's list of mass points.
in,out	*parent*	The new parent body; the body to which this body's root body is to be attached.

Definition at line 67 of file mass_attach.cc.

References jeod::MassPoint::attach(), attach_validate(), jeod::MassPoint::compute_state_wrt_pred(), jeod::MassPoint::detach(), find_mass_point(), jeod::generate_bad_point_message(), name, jeod::MassPointState::position, jeod::MassPointState::Q_parent_this, structure_point, and jeod::MassPointState::T_parent_this.

Referenced by attach_child().

8.1.3.7 attach_to() [2/2]

```
bool jeod::MassBody::attach_to (
            double offset_pstr_cstr_pstr[3],
            double T_pstr_cstr[3][3],
            MassBody & parent )  [virtual]
```

Attach this mass body's root body as a child of the specified mass body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with respect to the parent body's structural axes as specified.

Returns

Success indicator: true=success, false=attachment not performed.

Parameters

in	*offset_pstr_cstr_pstr*	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the new parent body. Units: M
in	*T_pstr_cstr*	Transformation matrix from the new parent body's structural frame to this body's structural frame.
in,out	*parent*	The new parent body; the body to which this body's root body is to be attached.

Definition at line 151 of file mass_attach.cc.

References jeod::MassBodyMessages::attach_info, attach_root_body(), attach_validate(), jeod::MassPoint::compute_state_wrt_pred(), get_root_body(), get_root_body_internal(), jeod::MassBodyMessages::invalid_attach, links, name, jeod::MassPointState::position, structure_point, and jeod::MassPointState::T_parent_this.

8.1.3.8 attach_update_properties()

```
void jeod::MassBody::attach_update_properties (
            const double offset_pstr_cstr_pstr[3],
            const double T_pstr_cstr[3][3],
            MassBody & child )  [protected], [virtual]
```

Set the relation between parent and child and update the mass properties.

Extensibility comments –

- This method is sent to the parent body of the attachment after the child body has established the logical connectivity between the parent body and child body.
- The generic purpose of this method is to establish the physical relation between parent and child and to update any physical properties that change as a result of the attachment.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The attachment is valid and logical connectivity has been established. Neither assumption is checked.

Parameters

in	*offset_pstr_cstr_pstr*	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the new parent body. Units: M
in	*T_pstr_cstr*	Transformation matrix from the new parent body's structural frame to this body's structural frame.
in,out	*child*	The child body; the body newly attached to this body.

Definition at line 494 of file mass_attach.cc.

References composite_properties, composite_wrt_pbdy, composite_wrt_pstr, jeod::MassPointState::compute_↩ transformation(), get_root_body_internal(), jeod::MassPointState::position, jeod::MassPointState::Q_parent_this, set_update_flag(), structure_point, jeod::MassPointState::T_parent_this, update_mass_properties(), jeod::Mass↩ PointState::update_orientation(), and jeod::MassPointState::update_point().

Referenced by attach_root_body().

8.1.3.9 attach_validate()

```
bool jeod::MassBody::attach_validate (
             const MassBody & parent,
             bool generate_message ) const  [virtual]
```

Validate whether the pending attachment is legal.

Extensibility comments –

- Overriding this method doesn't make a whole lot of sense.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The subject body, this, must be a root body. This is not checked.

Returns

Validity indicator

Parameters

in	*parent*	The new parent body; the body to which this body is to be attached.
in	*generate_message*	Generate message if invalid?

Definition at line 321 of file mass_attach.cc.

References attach_validate_child(), and attach_validate_parent().

Referenced by attach_root_body(), and attach_to().

8.1.3.10 attach_validate_child()

```
bool jeod::MassBody::attach_validate_child (
             const MassBody & child,
             bool generate_message ) const  [protected], [virtual]
```

Validate whether the pending attachment is legal from a physical point of view.

Extensibility comments –

- This method determines whether invoking attach_update_properties makes sense.
- This is a free pass for a MassBody.

Returns

Validity indicator

Parameters

in	*child*	The child body; the body to be attached to this body.
in	*generate_message*	Generate message if invalid?

Definition at line 405 of file mass_attach.cc.

References dyn_manager, get_root_body(), jeod::MassBodyMessages::invalid_attach, and name.

Referenced by attach_validate().

8.1.3.11 attach_validate_parent()

```
bool jeod::MassBody::attach_validate_parent (
            const MassBody & parent,
            bool generate_message ) const  [protected], [virtual]
```

Validate whether the pending attachment is legal from a connectivity point of view.

Extensibility comments –

- This method determines whether invoking attach_establish_links makes sense.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The subject body, this, must be a root body. This is not checked.

Returns

Validity indicator

Parameters

in	*parent*	The new parent body; the body to which this body is to be attached.
in	*generate_message*	Generate message if invalid?

Definition at line 350 of file mass_attach.cc.

References dyn_manager, get_root_body(), jeod::MassBodyMessages::invalid_attach, and name.

Referenced by attach_validate().

8.1.3.12 calc_composite_cm()

```
void jeod::MassBody::calc_composite_cm ( )  [protected]
```

Calculate the aggregate cm for a composite MassBody.

Assumptions and Limitations

- Rigid bodies
- Must calculate from bottom to top of tree for meaningful results

Definition at line 45 of file mass_calc_composite_cm.cc.

References composite_properties, composite_wrt_pstr, core_properties, jeod::MassProperties::inverse_mass, links, jeod::MassProperties::mass, and jeod::MassPointState::position.

Referenced by update_mass_properties().

8.1.3.13 calc_composite_inertia()

```
void jeod::MassBody::calc_composite_inertia ( )  [protected]
```

Calculate the aggregate inertia matrix for a composite MassBody.

Assumptions and Limitations

- Rigid Bodies
- Tree attachment structure
- Must calculate from bottom to top of tree for meaningful results

Definition at line 45 of file mass_calc_composite_inertia.cc.

References composite_properties, composite_wrt_pbdy, compute_point_mass_inertia(), core_properties, core↩_wrt_composite, jeod::MassProperties::inertia, links, jeod::MassProperties::mass, jeod::MassPointState::position, and jeod::MassPointState::T_parent_this.

Referenced by update_mass_properties().

8.1.3.14 compute_point_mass_inertia()

```
void jeod::MassBody::compute_point_mass_inertia (
              double mass,
              const double r_pt[3],
              double inertia[3][3] )  [static]
```

Compute the inertia tensor of a point mass.

Parameters

in	*mass*	Mass of point mass Units: kg
in	*r_pt*	Vector to point mass Units: M
out	*inertia*	Inertia tensor Units: kgM2

Definition at line 41 of file mass_point_mass_inertia.cc.

Referenced by calc_composite_inertia(), and jeod::MassPropertiesInit::initialize_mass_properties().

8.1.3.15 detach() [1/2]

```
bool jeod::MassBody::detach (
            MassBody & mass_body )  [virtual]
```

Detach the two bodies, 'this' and the argument body, from each other such such that the detachment occurs between the superior body and the immediate child along the path from the inferior body to the superior body.

Returns true to indicate success, false to indicate failure.

Assumptions and Limitations

- Instances of more derived classes, with presumably more involved dynamics, are situated higher in the mass tree than are more basic instances. For example, a simple MassBody can be a child of a DynBody, but not the other way around.
- The attachment in the mass tree between the immediate child and the superior body is assumed to reflect a real physical attachment.

Returns

Success flag

Parameters

in,out	*mass_body*	The other body

Definition at line 63 of file mass_detach.cc.

References detach(), jeod::MassBodyMessages::invalid_detach, links, and name.

Referenced by detach().

8.1.3.16 detach() [2/2]

```
bool jeod::MassBody::detach ( )  [virtual]
```

Detach a mass body from its immediate parent.

Returns true to indicate success, false to indicate failure.

Assumptions and Limitations

- Instances of more derived classes, with presumably more involved dynamics, are situated higher in the mass tree than are more basic instances. For example, a simple MassBody can be a child of a DynBody, but not the other way around.
- The attachment in the mass tree between the immediate child and the superior body is assumed to reflect a real physical attachment.

Returns

Success flag

Definition at line 133 of file mass_detach.cc.

References detach_sever_links(), detach_update_properties(), detach_validate(), dyn_owner, DynBody, and links.

Referenced by ∼MassBody().

8.1.3.17 detach_sever_links()

```
void jeod::MassBody::detach_sever_links (
            MassBody & parent )  [protected], [virtual]
```

Break the logical connectivity between parent and child.

Extensibility comments –

- This method is invoked before the updating the parent/child states.
- The generic purpose of this method is to sever all connectivity links between parent and child, in terms of the child class.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The detachment is valid; not checked.

Parameters

in,out	*parent*	The parent body; the body from which this body is to be detached.

Definition at line 292 of file mass_detach.cc.

References links.

Referenced by detach().

8.1.3.18 detach_update_properties()

```
void jeod::MassBody::detach_update_properties (
            MassBody & child )  [protected], [virtual]
```

Update parent and child properties to reflect that they are detached.

Extensibility comments –

- This method is sent to the parent body of the detachment after the child body has severed the logical connectivity between the parent body and child body.
- The generic purpose of this method is to update any physical properties that change as a result of the detachment.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The detachment is valid and logical connectivity has been severed. Neither assumption is checked.

Parameters

in,out	*child*	The child body; the body newly detached from this body.

Definition at line 315 of file mass_detach.cc.

References composite_properties, composite_wrt_pbdy, composite_wrt_pstr, compute_inverse_inertia, get_root_body_internal(), jeod::MassProperties::inertia, jeod::MassPoint::initialize_mass_point(), jeod::MassProperties::inverse_inertia, jeod::MassProperties::mass, set_update_flag(), structure_point, and update_mass_properties().

Referenced by detach().

8.1.3.19 detach_validate()

```
bool jeod::MassBody::detach_validate (
            const MassBody * parent,
            bool generate_message ) const  [virtual]
```

Validate whether the pending detachment is legal.

Extensibility comments –

- Overriding this method doesn't make a whole lot of sense.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Assumptions and Limitations

- The subject body, this, must be a root body. This is not checked.

Returns

Validity indicator

Parameters

in	*parent*	The parent body; the body from which this body is to be detached.
in	*generate_message*	Generate message if invalid?

Definition at line 184 of file mass_detach.cc.

References detach_validate_child(), and detach_validate_parent().

Referenced by detach().

8.1.3.20 detach_validate_child()

```
bool jeod::MassBody::detach_validate_child (
            const MassBody & child,
            bool generate_message ) const  [protected], [virtual]
```

Validate whether the pending detachment is legal from a mass tree point of view.

Extensibility comments –

- This method determines whether invoking detach_update_properties makes sense.

  Returns

  Validity indicator

Parameters

in	*child*	The child body; the body to be detached from this body.
in	*generate_message*	Generate message if invalid?

Definition at line 257 of file mass_detach.cc.

References dyn_manager, jeod::MassBodyMessages::invalid_detach, and name.

Referenced by detach_validate().

8.1.3.21 detach_validate_parent()

```
bool jeod::MassBody::detach_validate_parent (
            const MassBody * parent,
            bool generate_message ) const  [protected], [virtual]
```

Validate whether the pending detachment is legal from a connectivity point of view.

Extensibility comments –

- This method determines whether invoking detach_sever_links makes sense.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

  Returns

  Validity indicator

Parameters

in	*parent*	The parent body; the body from which this body is to be detached.
in	*generate_message*	Generate message if invalid?

Definition at line 210 of file mass_detach.cc.

References dyn_manager, jeod::MassBodyMessages::invalid_detach, and name.

Referenced by detach_validate().

8.1.3.22 find_mass_point()

```
const MassPoint * jeod::MassBody::find_mass_point (
            const std::string & pt_name ) const
```

Find the mass point with the given name.

Returns

Mass point

Parameters

in	*pt_name*	mass point name

Definition at line 307 of file mass.cc.

References mass_points, and name.

Referenced by add_mass_point(), and attach_to().

8.1.3.23 get_mass_properties_initialized()

```
bool jeod::MassBody::get_mass_properties_initialized ( )  [inline]
```

Definition at line 134 of file mass.hh.

References mass_properties_initialized.

8.1.3.24 get_parent_body()

```
const MassBody * jeod::MassBody::get_parent_body ( ) const  [virtual]
```

Returns the MassBody's parent body, as a const pointer.

Returns

Pointer to parent body

Definition at line 224 of file mass.cc.

References links.

8.1.3.25 get_parent_body_internal()

```
MassBody * jeod::MassBody::get_parent_body_internal ( )  [protected], [virtual]
```

Returns the MassBody's parent body, as a non-const pointer.

Returns

Pointer to parent body

Definition at line 233 of file mass.cc.

References links.

8.1.3.26 get_root_body()

```
const MassBody * jeod::MassBody::get_root_body ( ) const  [virtual]
```

Finds & returns root of current MassBody's tree.

Returns

Pointer to root body

Definition at line 242 of file mass.cc.

References links.

Referenced by attach_to(), attach_validate_child(), attach_validate_parent(), and print_tree().

8.1.3.27 get_root_body_internal()

```
MassBody * jeod::MassBody::get_root_body_internal ( )  [protected], [virtual]
```

Finds & returns root of current MassBody's tree.

Returns

Pointer to root body

Definition at line 251 of file mass.cc.

References links.

Referenced by attach_to(), attach_update_properties(), detach_update_properties(), and reattach().

8.1.3.28 initialize_mass()

```
void jeod::MassBody::initialize_mass (
            const MassPropertiesInit & properties,
            const std::vector< MassPointInit *> & points )
```

Initialize a MassBody object.

Parameters

in	*properties*	Core mass ppty specs
in	*points*	Mass point specs

Definition at line 191 of file mass.cc.

References add_mass_point(), composite_properties, core_properties, dyn_owner, jeod::MassPropertiesInit::initialize_mass_properties(), mass_properties_initialized, jeod::MassPointState::Q_parent_this, set_update_flag(), and jeod::MassPointState::T_parent_this.

8.1.3.29 is_progeny_of()

```
bool jeod::MassBody::is_progeny_of (
            const MassBody & test_body ) const
```

Return true if this MassBody is an offspring of provided one, false if not.

Returns

Is this offspring of test_body?

Parameters

in	*test_body*	Other MassBody

Definition at line 262 of file mass.cc.

References links.

8.1.3.30 mass_points_size()

```
size_t jeod::MassBody::mass_points_size ( ) const
```

Return the number of mass points for this body.

Returns

Mass point

Definition at line 297 of file mass.cc.

References mass_points.

8.1.3.31 operator=()

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

8.1.3.32 print_body()

```
void jeod::MassBody::print_body (
             FILE * file_ptr,
             int levels ) const
```

Recursively print out the data in this and its children to given file.

Parameters

in,out	*file_ptr*	Output file stream
in,out	*levels*	Max desired recursion level

Definition at line 44 of file mass_print_body.cc.

References composite_properties, core_properties, jeod::MassProperties::inertia, jeod::MassProperties::inverse↩_inertia, jeod::MassProperties::inverse_mass, links, jeod::MassProperties::mass, name, jeod::MassPointState↩::position, print_body(), structure_point, and jeod::MassPointState::T_parent_this.

Referenced by print_body(), and print_tree().

8.1.3.33 print_tree()

```
void jeod::MassBody::print_tree (
             const char * file_name,
             int levels ) const
```

Print out the MassTree members' data to a given file.

Parameters

in	*file_name*	Desired output file name
in,out	*levels*	Max desired recursion level

Definition at line 47 of file mass_print_tree.cc.

References get_root_body(), jeod::MassBodyMessages::io_error, and print_body().

8.1.3.34 reattach()

```
bool jeod::MassBody::reattach (
             double offset[3],
             double T_pstr_cstr[3][3] )  [virtual]
```

Re-attach a child MassBody to a parent MassBody.

Assumptions and Limitations

- Rigid Bodies
- Tree attachment structure
- Child is known to be a root or atomic body
- Re-establishing an attachment that previously existed

Parameters

in	*offset*	Desired offset from parent struct frame to attached child's struct frame, expressed in parent struct frame Units: M
in	*T_pstr_cstr*	Desired transformation matrix from parent struct frame to attached child's struct frame

Definition at line 52 of file mass_reattach.cc.

References composite_properties, composite_wrt_pbdy, composite_wrt_pstr, jeod::MassPointState::compute_transformation(), get_root_body_internal(), jeod::MassBodyMessages::invalid_node, links, name, jeod::MassPointState::position, jeod::MassPointState::Q_parent_this, set_update_flag(), structure_point, jeod::MassPointState::T_parent_this, update_mass_properties(), jeod::MassPointState::update_orientation(), and jeod::MassPointState::update_point().

8.1.3.35 set_name()

```
void jeod::MassBody::set_name (
            std::string name_in )  [inline]
```

Definition at line 140 of file mass.hh.

References name.

8.1.3.36 set_update_flag()

```
void jeod::MassBody::set_update_flag ( )
```

Flag mass bodies from the current body on up the mass tree as in need of mass property updates.

Definition at line 284 of file mass.cc.

References links.

Referenced by attach_update_properties(), detach_update_properties(), initialize_mass(), reattach(), and ∼MassBody().

8.1.3.37 update_mass_properties()

```
void jeod::MassBody::update_mass_properties ( )  [virtual]
```

Update composite mass properties for a mass tree marked for update.

The properties are updated from the bottom up to generate correct results.

Assumptions and Limitations

- Rigid bodies
- Tree attachment structure

Definition at line 50 of file mass_update.cc.

References calc_composite_cm(), calc_composite_inertia(), composite_properties, composite_wrt_pbdy, composite_wrt_pstr, compute_inverse_inertia, core_properties, core_wrt_composite, jeod::MassProperties::inertia, jeod::MassProperties::inverse_inertia, jeod::MassProperties::inverse_mass, links, jeod::MassProperties::mass, needs_update, jeod::MassPointState::position, structure_point, jeod::MassPointState::T_parent_this, and update_mass_properties().

Referenced by attach_update_properties(), detach_update_properties(), reattach(), and update_mass_properties().

8.1.4 Friends And Related Function Documentation

8.1.4.1 DynBody

```
friend class DynBody  [friend]
```

Definition at line 115 of file mass.hh.

Referenced by detach().

8.1.4.2 init_attrjeod__MassBody

```
void init_attrjeod__MassBody ( )  [friend]
```

8.1.4.3 InputProcessor

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

Definition at line 114 of file mass.hh.

8.1.4.4 MassBodyLinks

```
friend class MassBodyLinks  [friend]
```

Definition at line 114 of file mass.hh.

8.1.5 Field Documentation

8.1.5.1 composite_properties

```
MassProperties jeod::MassBody::composite_properties
```

The mass properties of this body, including child bodies.

The composite_properties are core_properties are synonymous for atomic bodies. The composite_properties is a child of the structure_point.trick_units(--)

Definition at line 227 of file mass.hh.

Referenced by attach_update_properties(), calc_composite_cm(), calc_composite_inertia(), detach_update_↩ properties(), initialize_mass(), MassBody(), print_body(), reattach(), and update_mass_properties().

8.1.5.2 composite_wrt_pbdy

```
MassPoint jeod::MassBody::composite_wrt_pbdy  [protected]
```

The location and orientation of the composite CoM and body frame with respect to the parent body's composite CoM and body frame.

trick_units(–)

Definition at line 356 of file mass.hh.

Referenced by attach_establish_links(), attach_update_properties(), calc_composite_inertia(), detach_update_↩ properties(), MassBody(), reattach(), and update_mass_properties().

8.1.5.3 composite_wrt_pstr

```
MassPoint jeod::MassBody::composite_wrt_pstr  [protected]
```

The location and orientation of the composite CoM and body frame with respect to the parent body's structural frame.

trick_units(–)

Definition at line 350 of file mass.hh.

Referenced by attach_establish_links(), attach_update_properties(), calc_composite_cm(), detach_update_↩ properties(), reattach(), and update_mass_properties().

8.1.5.4 compute_inverse_inertia

```
bool jeod::MassBody::compute_inverse_inertia {}
```

When clear, the inverse of the inertia tensor is not calculated.

The inverse of the inertia tensor is only calculated if this flag is set and the composite mass is positive.trick_units(–)

Definition at line 234 of file mass.hh.

Referenced by detach_update_properties(), and update_mass_properties().

8.1.5.5 core_properties

```
MassProperties jeod::MassBody::core_properties
```

The mass properties of this body alone, excluding child bodies.

The core_properties object is a child of the structure_point.trick_units(–)

Definition at line 220 of file mass.hh.

Referenced by calc_composite_cm(), calc_composite_inertia(), initialize_mass(), MassBody(), print_body(), and update_mass_properties().

8.1.5.6 core_wrt_composite

```
MassPoint jeod::MassBody::core_wrt_composite  [protected]
```

The location and orientation of the composite CoM and body frame with respect to the parent body's composite CoM and body frame.

trick_units(–)

Definition at line 344 of file mass.hh.

Referenced by calc_composite_inertia(), MassBody(), and update_mass_properties().

8.1.5.7 dyn_manager

```
BaseDynManager* jeod::MassBody::dyn_manager {}  [protected]
```

The dynamics manager for the simulation.

trick_units(–)

Definition at line 308 of file mass.hh.

Referenced by attach_validate_child(), attach_validate_parent(), detach_validate_child(), and detach_validate_↩
parent().

8.1.5.8 dyn_owner

```
DynBody* const jeod::MassBody::dyn_owner {}
```

Indicates this body belongs to a DynBody.

Many JEOD functions manipulate MassBody properties with the implementation depending/varying upon whether the MassBody refers to a dynamic body.trick_units(–)

Definition at line 249 of file mass.hh.

Referenced by detach(), and initialize_mass().

8.1.5.9 links

```
MassBodyLinks jeod::MassBody::links [protected]
```

Linkage to rest of mass tree.

Programmatic interfaces:

- MassBodyLinks provides accessors to the parent and root and provides methods to attach, detach links (and hence bodies).
- This class provides accessors to the same.
- Various iterators provide the ability to iterate over child bodies and up the parent chain.trick_units(–)

Definition at line 326 of file mass.hh.

Referenced by attach_establish_links(), attach_root_body(), attach_to(), calc_composite_cm(), calc_composite↩_inertia(), detach(), detach_sever_links(), get_parent_body(), get_parent_body_internal(), get_root_body(), get_↩root_body_internal(), is_progeny_of(), print_body(), reattach(), set_update_flag(), update_mass_properties(), and ∼MassBody().

8.1.5.10 mass_points

```
std::list<MassPoint *> jeod::MassBody::mass_points [protected]
```

List of points associated with this mass body.

NOTE WELL: The MassBody manages the memory associated with the contents of this list.

Definition at line 338 of file mass.hh.

Referenced by add_mass_point(), find_mass_point(), mass_points_size(), and ∼MassBody().

8.1.5.11 mass_properties_initialized

```
bool jeod::MassBody::mass_properties_initialized {} [protected]
```

Initialized flag.

Indicates whether the initialize_mass method has been executed, specifically the initialize_mass_properties method from the initialize_mass method.trick_units(–)

Definition at line 315 of file mass.hh.

Referenced by get_mass_properties_initialized(), and initialize_mass().

8.1.5.12 name

```
NamedItem jeod::MassBody::name
```

Body name.

trick_units(–)

Definition at line 214 of file mass.hh.

Referenced by add_mass_point(), attach_root_body(), attach_to(), attach_validate_child(), attach_validate_↩ parent(), detach(), detach_validate_child(), detach_validate_parent(), find_mass_point(), print_body(), reattach(), set_name(), and ∼MassBody().

8.1.5.13 needs_update

```
bool jeod::MassBody::needs_update {}  [protected]
```

When set, indicates that an update to the composite mass is needed.

trick_units(–)

Definition at line 331 of file mass.hh.

Referenced by update_mass_properties().

8.1.5.14 structure_point

```
MassPoint jeod::MassBody::structure_point
```

The location and orientation of the structural frame with respect to the parent body's structural frame.

Attaching one MassBody to some other MassBody makes the attaching MassBody's structure_point a child of the attachee's structure_point.trick_units(–)

Definition at line 242 of file mass.hh.

Referenced by add_mass_point(), attach_establish_links(), attach_to(), attach_update_properties(), detach_↩ update_properties(), MassBody(), print_body(), reattach(), and update_mass_properties().

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

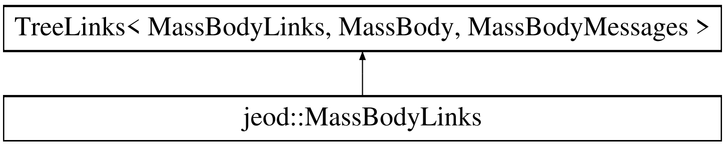
- mass.hh
- mass.cc
- mass_attach.cc
- mass_calc_composite_cm.cc
- mass_calc_composite_inertia.cc
- mass_detach.cc
- mass_point_mass_inertia.cc
- mass_print_body.cc
- mass_print_tree.cc
- mass_reattach.cc
- mass_update.cc

8.2 jeod::MassBodyLinks Class Reference

Encapsulates the links between mass bodies.

```
#include <mass_body_links.hh>
```

Inheritance diagram for jeod::MassBodyLinks:



Public Member Functions

- MassBodyLinks (MassBody &container_in)

  Non-default constructor.
- MassBodyLinks ()=delete
- MassBodyLinks (const MassBodyLinks &)=delete
- void operator= (const MassBodyLinks &)=delete
- ∼MassBodyLinks () override=default

  Destructor.

Static Private Attributes

- static const unsigned int default_path_size = 8

Friends

- class InputProcessor
- void init_attrjeod__MassBodyLinks ()

8.2.1 Detailed Description

Encapsulates the links between mass bodies.

Definition at line 81 of file mass_body_links.hh.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 MassBodyLinks() [1/3]

```
jeod::MassBodyLinks::MassBodyLinks (
            MassBody & container_in )  [inline], [explicit]
```

Non-default constructor.

Parameters

container← _in	The MassBody object that contains this object.

Definition at line 89 of file mass_body_links.hh.

8.2.2.2 MassBodyLinks() [2/3]

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

8.2.2.3 MassBodyLinks() [3/3]

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

8.2.2.4 ∼MassBodyLinks()

```
jeod::MassBodyLinks::∼MassBodyLinks ( )  [override], [default]
```

Destructor.

8.2.3 Member Function Documentation

8.2.3.1 operator=()

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

8.2.4 Friends And Related Function Documentation

8.2.4.1 init_attrjeod__MassBodyLinks

```
void init_attrjeod__MassBodyLinks ( )  [friend]
```

8.2.4.2 InputProcessor

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

Definition at line 83 of file mass_body_links.hh.

8.2.5 Field Documentation

8.2.5.1 default_path_size

```
const unsigned int jeod::MassBodyLinks::default_path_size = 8 [static], [private]
```

Definition at line 106 of file mass_body_links.hh.

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

- mass_body_links.hh

8.3 jeod::MassBodyMessages Class Reference

Specify the message IDs used in the MassBody model.

```
#include <mass_messages.hh>
```

Public Member Functions

- MassBodyMessages ()=delete
- MassBodyMessages (const MassBodyMessages &)=delete
- MassBodyMessages & operator= (const MassBodyMessages &)=delete

Static Public Attributes

- static const char ∗ attach_info = "dynamics/mass/" "attach_info"

  Issued to provide information regarding an attachment.
- static const char ∗ invalid_attach = "dynamics/mass/" "invalid_attach"

  Issued when an attachment cannot be performed as requested.
- static const char ∗ invalid_detach = "dynamics/mass/" "invalid_detach"

  Issued when a detachment cannot be performed as requested.
- static const char ∗ invalid_node = "dynamics/mass/" "invalid_node"

  Issued when a node does not have expected linkages.
- static const char ∗ invalid_name = "dynamics/mass/" "invalid_name"

  Issued when a name is invalid – NULL, empty, a duplicate, ...
- static const char ∗ invalid_enum = "dynamics/mass/" "invalid_enum"

  Issued when a enum value is not one of the enumerated values.
- static const char ∗ io_error = "dynamics/mass/" "io_error"

  Issued when an I/O error occurs.
- static const char ∗ internal_error = "dynamics/mass/" "internal_error"

  Error issued when some internal error occurred.

Friends

- class InputProcessor
- void init_attrjeod__MassBodyMessages ()

8.3.1 Detailed Description

Specify the message IDs used in the MassBody model.

Assumptions and Limitations

- This is a complete catalog of all the messages sent by the MassBody model.
- This is not an exhaustive list of all the things that can go awry.

Definition at line 81 of file mass_messages.hh.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 MassBodyMessages() [1/2]

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

8.3.2.2 MassBodyMessages() [2/2]

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

8.3.3 Member Function Documentation

8.3.3.1 operator=()

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

8.3.4 Friends And Related Function Documentation

8.3.4.1 init_attrjeod__MassBodyMessages

```
void init_attrjeod__MassBodyMessages ( )  [friend]
```

8.3.4.2 InputProcessor

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

Definition at line 83 of file mass_messages.hh.

8.3.5 Field Documentation

8.3.5.1 attach_info

```
char const * jeod::MassBodyMessages::attach_info = "dynamics/mass/" "attach_info"  [static]
```

Issued to provide information regarding an attachment.

trick_units(–)

Definition at line 88 of file mass_messages.hh.

Referenced by jeod::MassBody::attach_to().

8.3.5.2 internal_error

```
char const * jeod::MassBodyMessages::internal_error = "dynamics/mass/" "internal_error"  [static]
```

Error issued when some internal error occurred.

These errors should never happen.trick_units(–)

Definition at line 124 of file mass_messages.hh.

8.3.5.3 invalid_attach

```
char const * jeod::MassBodyMessages::invalid_attach = "dynamics/mass/" "invalid_attach"  [static]
```

Issued when an attachment cannot be performed as requested.

trick_units(–)

Definition at line 93 of file mass_messages.hh.

Referenced by jeod::MassBody::attach_root_body(), jeod::MassBody::attach_to(), jeod::MassBody::attach_↩ validate_child(), jeod::MassBody::attach_validate_parent(), and jeod::generate_bad_point_message().

8.3.5.4 invalid_detach

```
char const * jeod::MassBodyMessages::invalid_detach = "dynamics/mass/" "invalid_detach"  [static]
```

Issued when a detachment cannot be performed as requested.

trick_units(–)

Definition at line 98 of file mass_messages.hh.

Referenced by jeod::MassBody::detach(), jeod::MassBody::detach_validate_child(), jeod::MassBody::detach_validate_parent(), and jeod::MassBody::∼MassBody().

8.3.5.5 invalid_enum

```
char const * jeod::MassBodyMessages::invalid_enum = "dynamics/mass/" "invalid_enum"  [static]
```

Issued when a enum value is not one of the enumerated values.

trick_units(–)

Definition at line 113 of file mass_messages.hh.

Referenced by jeod::MassPointInit::initialize_mass_point(), and jeod::MassPropertiesInit::initialize_mass_properties().

8.3.5.6 invalid_name

```
char const * jeod::MassBodyMessages::invalid_name = "dynamics/mass/" "invalid_name"  [static]
```

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

trick_units(–)

Definition at line 108 of file mass_messages.hh.

Referenced by jeod::MassBody::add_mass_point().

8.3.5.7 invalid_node

```
char const * jeod::MassBodyMessages::invalid_node = "dynamics/mass/" "invalid_node"  [static]
```

Issued when a node does not have expected linkages.

trick_units(–)

Definition at line 103 of file mass_messages.hh.

Referenced by jeod::MassPoint::compute_pred_rel_state(), jeod::MassPoint::compute_relative_state(), jeod::MassPoint::compute_state_wrt_pred(), and jeod::MassBody::reattach().

8.3.5.8 io_error

```
char const * jeod::MassBodyMessages::io_error = "dynamics/mass/" "io_error"  [static]
```

Issued when an I/O error occurs.

trick_units(–)

Definition at line 118 of file mass_messages.hh.

Referenced by jeod::MassBody::print_tree().

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

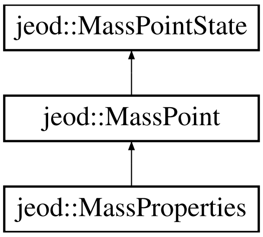
- mass_messages.hh
- mass_messages.cc

8.4 jeod::MassPoint Class Reference

Adds tree linkages and a name to a MassPointState.

`#include <mass_point.hh>`

Inheritance diagram for jeod::MassPoint:



Public Member Functions

- MassPoint ()

  Default constructor.
- ∼MassPoint () override

  Destroy a MassPoint object.
- MassPoint (const MassPoint &)=delete
- MassPoint & operator= (const MassPoint &)=delete
- void initialize_mass_point () override

  Initialize a mass point.
- void set_name (std::string name_in)
- std::string get_name () const
- virtual void compute_relative_state (const MassPoint &ref_point, MassPointState &rel_state) const

  Compute the state of this point with respect to some reference point, which may be any point in the same tree as this point.
- virtual void compute_state_wrt_pred (const MassPoint &ref_point, MassPointState &rel_state) const

  Compute the state of this point with respect to some reference point, which must be at or above this point via the parent links.

- virtual void compute_state_wrt_pred (unsigned int ref_point_index, MassPointState &rel_state) const

  Compute the state of this point with respect to some reference point, which must be at or above this point via the parent links.
- virtual void compute_pred_rel_state (const MassPoint &ref_point, MassPointState &rel_state) const

  Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which must be a predecessor of the invoking frame.
- virtual void compute_pred_rel_state (unsigned int ref_point_index, MassPointState &rel_state) const

  Compute the state of some reference point with respect to this point.
- const MassPoint ∗ find_last_common_node (const MassPoint &ref_point) const

  Each mass point has a path from the root of the mass point tree to the point in question.

Protected Member Functions

- int find_last_common_index (const MassPoint &ref_point) const

  Each mass point has a path from the root of the mass point tree to the point in question.

Protected Attributes

- NamedItem name

  The name of the mass point.
- MassPointLinks links

  Linkage to rest of mass tree.

Private Member Functions

- void attach (MassPoint &parent)

  Attach a mass point to another.
- void detach ()

  Detach a mass point from its parent.

Friends

- class InputProcessor
- class MassPointLinks
- class MassBody
- void init_attrjeod__MassPoint ()

Additional Inherited Members

8.4.1 Detailed Description

Adds tree linkages and a name to a MassPointState.

Definition at line 86 of file mass_point.hh.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 MassPoint() [1/2]

```
jeod::MassPoint::MassPoint ( )  [inline]
```

Default constructor.

Definition at line 114 of file mass_point.hh.

8.4.2.2 ∼MassPoint()

```
jeod::MassPoint::∼MassPoint ( )  [override]
```

Destroy a MassPoint object.

Definition at line 58 of file mass_point.cc.

References links.

8.4.2.3 MassPoint() [2/2]

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

8.4.3 Member Function Documentation

8.4.3.1 attach()

```
void jeod::MassPoint::attach (
             MassPoint & parent )  [inline], [private]
```

Attach a mass point to another.

Assumptions and Limitations

- This method only addresses the linkages. Some external agent must address the physical relation.

Parameters

in,out	*parent*	parent node

Definition at line 226 of file mass_point.hh.

Referenced by jeod::MassBody::add_mass_point(), jeod::MassBody::attach_establish_links(), jeod::MassBody::attach_to(), and jeod::MassBody::MassBody().

8.4.3.2 compute_pred_rel_state() [1/2]

```
void jeod::MassPoint::compute_pred_rel_state (
             const MassPoint & ref_point,
             MassPointState & rel_state ) const  [virtual]
```

Compute the complete state of the invoking reference frame with respect to the supplied reference frame, which *must* be a predecessor of the invoking frame.

Assumptions and Limitations

- The predecessor frame is a predecessor.

Parameters

in	*ref_point*	The point with respect to which the state is to be expressed
out	*rel_state*	The relative state

Definition at line 231 of file mass_point.cc.

References jeod::MassBodyMessages::invalid_node, and links.

Referenced by compute_relative_state().

8.4.3.3 compute_pred_rel_state() [2/2]

```
void jeod::MassPoint::compute_pred_rel_state (
             unsigned int ref_point_index,
             MassPointState & rel_state ) const  [virtual]
```

Compute the state of some reference point with respect to this point.

The reference point must be at or above this point via the parent links.

Parameters

in	*ref_point_index*	Reference point index
out	*rel_state*	Relative state

Definition at line 258 of file mass_point.cc.

References jeod::MassPointState::initialize_mass_point(), links, jeod::MassPointState::position, jeod::MassPointState::Q_parent_this, and jeod::MassPointState::T_parent_this.

8.4.3.4 compute_relative_state()

```
void jeod::MassPoint::compute_relative_state (
            const MassPoint & ref_point,
            MassPointState & rel_state ) const  [virtual]
```

Compute the state of this point with respect to some reference point, which may be any point in the same tree as this point.

Parameters

in	*ref_point*	The point with respect to which the state is to be expressed
out	*rel_state*	The relative state

Definition at line 87 of file mass_point.cc.

References compute_pred_rel_state(), compute_state_wrt_pred(), jeod::MassPointState::decr_left(), find_last_common_index(), jeod::MassPointState::initialize_mass_point(), jeod::MassBodyMessages::invalid_node, and links.

8.4.3.5 compute_state_wrt_pred() [1/2]

```
void jeod::MassPoint::compute_state_wrt_pred (
            const MassPoint & ref_point,
            MassPointState & rel_state ) const  [virtual]
```

Compute the state of this point with respect to some reference point, which must be at or above this point via the parent links.

Parameters

in	*ref_point*	Reference point
out	*rel_state*	Point state wrt ref. pt.

Definition at line 155 of file mass_point.cc.

References jeod::MassBodyMessages::invalid_node, and links.

Referenced by jeod::MassBody::attach_to(), and compute_relative_state().

8.4.3.6 compute_state_wrt_pred() [2/2]

```
void jeod::MassPoint::compute_state_wrt_pred (
            unsigned int ref_point_index,
            MassPointState & rel_state ) const  [virtual]
```

Compute the state of this point with respect to some reference point, which must be at or above this point via the parent links.

Parameters

in	*ref_point_index*	Reference point index
out	*rel_state*	Point state wrt ref. pt.

Definition at line 181 of file mass_point.cc.

References jeod::MassPointState::initialize_mass_point(), links, jeod::MassPointState::position, jeod::MassPoint←State::Q_parent_this, and jeod::MassPointState::T_parent_this.

8.4.3.7 detach()

```
void jeod::MassPoint::detach ( )  [inline], [private]
```

Detach a mass point from its parent.

Assumptions and Limitations

- This method only addresses the linkages. Some external agent must address the physical relation.

Definition at line 238 of file mass_point.hh.

Referenced by jeod::MassBody::attach_to(), initialize_mass_point(), and jeod::MassBody::∼MassBody().

8.4.3.8 find_last_common_index()

```
int jeod::MassPoint::find_last_common_index (
            const MassPoint & ref_point ) const  [inline], [protected]
```

Each mass point has a path from the root of the mass point tree to the point in question.

The paths for two mass points will have some initial sequence of common nodes. Find the index number of this last element in this sequence.

Returns

Last common node

Parameters

in	*ref_point*	Other point

Definition at line 192 of file mass_point.hh.

Referenced by compute_relative_state().

8.4.3.9 find_last_common_node()

```
const MassPoint * jeod::MassPoint::find_last_common_node (
             const MassPoint & frame ) const  [inline]
```

Each mass point has a path from the root of the mass point tree to the point in question.

The paths for two mass points will have some initial sequence of common nodes. Find the last element in this sequence.

Returns

Last common node

Parameters

in	*frame*	Other point

Definition at line 205 of file mass_point.hh.

8.4.3.10 get_name()

```
std::string jeod::MassPoint::get_name ( ) const  [inline]
```

Definition at line 135 of file mass_point.hh.

References name.

8.4.3.11 initialize_mass_point()

```
void jeod::MassPoint::initialize_mass_point ( )  [override], [virtual]
```

Initialize a mass point.

Reimplemented from jeod::MassPointState.

Definition at line 75 of file mass_point.cc.

References detach(), and jeod::MassPointState::initialize_mass_point().

Referenced by jeod::MassBody::detach_update_properties().

8.4.3.12 operator=()

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

8.4.3.13 set_name()

```
void jeod::MassPoint::set_name (
             std::string name_in )  [inline]
```

Definition at line 129 of file mass_point.hh.

References name.

8.4.4 Friends And Related Function Documentation

8.4.4.1 init_attrjeod__MassPoint

```
void init_attrjeod__MassPoint ( )  [friend]
```

8.4.4.2 InputProcessor

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

Definition at line 88 of file mass_point.hh.

8.4.4.3 MassBody

```
friend class MassBody  [friend]
```

Definition at line 89 of file mass_point.hh.

8.4.4.4 MassPointLinks

```
friend class MassPointLinks  [friend]
```

Definition at line 88 of file mass_point.hh.

8.4.5 Field Documentation

8.4.5.1 links

`MassPointLinks jeod::MassPoint::links [protected]`

Linkage to rest of mass tree.

Programmatic interfaces:

- MassPointLinks provides accessors to the parent and root and provides methods to attach, detach links (and hence bodies).
- This class provides accessors to the same.
- Various iterators provide the ability to iterate over child bodies and up the parent chain.trick_units(–)

Definition at line 107 of file mass_point.hh.

Referenced by compute_pred_rel_state(), compute_relative_state(), compute_state_wrt_pred(), and ∼MassPoint().

8.4.5.2 name

`NamedItem jeod::MassPoint::name [protected]`

The name of the mass point.

trick_units(–)

Definition at line 96 of file mass_point.hh.

Referenced by jeod::MassBody::add_mass_point(), get_name(), and set_name().

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

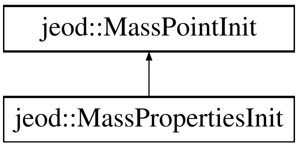
- mass_point.hh
- mass_point.cc

8.5 jeod::MassPointInit Class Reference

Contains data used to initialize a MassPoint.

`#include <mass_point_init.hh>`

Inheritance diagram for jeod::MassPointInit:



Public Types

- enum FrameSpec {
  StructToBody = 0, StructToCase = 0, StructToPoint = 0, StructToChild = 0,
  BodyToStruct = 1, CaseToStruct = 1, PointToStruct = 1, ChildToStruct = 1 }

  Specifies sense of the input point orientation data.

Public Member Functions

- MassPointInit ()
- virtual ∼MassPointInit ()=default
- void initialize_mass_point (MassPoint &mass_point) const

  Default constructor; constructs a MassPointInit object.
- void set_name (std::string name_in)

  Set the name.

Data Fields

- double position [3] {}

  Mass point location expressed in mass element structural coordinates.
- Orientation pt_orientation

  Mass point frame orientation specification.
- FrameSpec pt_frame_spec {StructToPoint}

  Indicates whether user orientation input defines the structure-to-body or body-to-structure transformation matrix.
- std::string name {""}

  Item name, with the following semantics for items that have a name:

Friends

- class InputProcessor
- void init_attrjeod__MassPointInit ()

8.5.1 Detailed Description

Contains data used to initialize a MassPoint.

Definition at line 87 of file mass_point_init.hh.

8.5.2 Member Enumeration Documentation

8.5.2.1 FrameSpec

```
enum jeod::MassPointInit::FrameSpec
```

Specifies sense of the input point orientation data.

The overloading of names is intentional.

Enumerator

StructToBody	Orientation specifies structure-to-body transform.
StructToCase	Orientation specifies structure-to-case transform.
StructToPoint	Orientation specifies structure-to-point transform.
StructToChild	Orientation specifies structure-to-child transform.
BodyToStruct	Orientation specifies body-to-structure transform.
CaseToStruct	Orientation specifies case-to-structure transform.
PointToStruct	Orientation specifies point-to-structure transform.
ChildToStruct	Orientation specifies child-to-structure transform.

Definition at line 94 of file mass_point_init.hh.

8.5.3 Constructor & Destructor Documentation

8.5.3.1 MassPointInit()

```
jeod::MassPointInit::MassPointInit ( )
```

Definition at line 51 of file mass_point_init.cc.

8.5.3.2 ∼MassPointInit()

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

8.5.4 Member Function Documentation

8.5.4.1 initialize_mass_point()

```
void jeod::MassPointInit::initialize_mass_point (
             MassPoint & mass_point ) const
```

Default constructor; constructs a MassPointInit object.

Parameters

out	*mass_point*	Point to initialize

Definition at line 60 of file mass_point_init.cc.

References jeod::MassBodyMessages::invalid_enum, PointToStruct, position, pt_frame_spec, pt_orientation, jeod::MassPointState::Q_parent_this, StructToPoint, jeod::MassPointState::T_parent_this, and jeod::MassPoint↩ State::update_point().

Referenced by jeod::MassBody::add_mass_point(), and jeod::MassPropertiesInit::initialize_mass_properties().

8.5.4.2 set_name()

```
void jeod::MassPointInit::set_name (
             std::string name_in )  [inline]
```

Set the name.

Definition at line 144 of file mass_point_init.hh.

8.5.5 Friends And Related Function Documentation

8.5.5.1 init_attrjeod__MassPointInit

```
void init_attrjeod__MassPointInit ( )  [friend]
```

8.5.5.2 InputProcessor

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

Definition at line 89 of file mass_point_init.hh.

8.5.6 Field Documentation

8.5.6.1 name

```
std::string jeod::MassPointInit::name {""}
```

Item name, with the following semantics for items that have a name:

- The mass point name will always be of the form "mass_name.point_name".
- If the supplied name does not begin with "mass_name.", this prefix will be applied to the supplied name in naming the mass point.

This can be left as the empty string for items that don't have a name.trick_units(–)

Definition at line 132 of file mass_point_init.hh.

Referenced by jeod::MassBody::add_mass_point().

8.5.6.2 position

```
double jeod::MassPointInit::position[3] {}
```

Mass point location expressed in mass element structural coordinates.

trick_units(m)

Definition at line 111 of file mass_point_init.hh.

Referenced by initialize_mass_point(), and jeod::MassPropertiesInit::MassPropertiesInit().

8.5.6.3 pt_frame_spec

```
FrameSpec jeod::MassPointInit::pt_frame_spec {StructToPoint}
```

Indicates whether user orientation input defines the structure-to-body or body-to-structure transformation matrix.

trick_units(--)

Definition at line 122 of file mass_point_init.hh.

Referenced by initialize_mass_point().

8.5.6.4 pt_orientation

```
Orientation jeod::MassPointInit::pt_orientation
```

Mass point frame orientation specification.

trick_units(--)

Definition at line 116 of file mass_point_init.hh.

Referenced by initialize_mass_point().

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

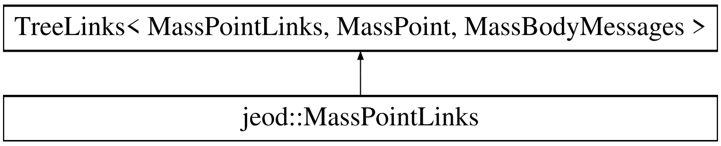
- mass_point_init.hh
- mass_point_init.cc

8.6 jeod::MassPointLinks Class Reference

Encapsulates the links between mass points.

```
#include <mass_point_links.hh>
```

Inheritance diagram for jeod::MassPointLinks:



Public Member Functions

- MassPointLinks (MassPoint &container_in)

  Non-default constructor.
- MassPointLinks ()=delete
- MassPointLinks (const MassPointLinks &)=delete
- void operator= (const MassPointLinks &)=delete
- ∼MassPointLinks () override=default

  Destructor.

Static Private Attributes

- static const unsigned int default_path_size = 8

Friends

- class InputProcessor
- void init_attrjeod__MassPointLinks ()

8.6.1 Detailed Description

Encapsulates the links between mass points.

Definition at line 81 of file mass_point_links.hh.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 MassPointLinks() [1/3]

```
jeod::MassPointLinks::MassPointLinks (
            MassPoint & container_in )  [inline], [explicit]
```

Non-default constructor.

Parameters

container← _in	The MassPoint object that contains this object.

Definition at line 89 of file mass_point_links.hh.

8.6.2.2 MassPointLinks() [2/3]

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

8.6.2.3 MassPointLinks() [3/3]

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

8.6.2.4 ∼MassPointLinks()

```
jeod::MassPointLinks::∼MassPointLinks ( )  [override], [default]
```

Destructor.

8.6.3 Member Function Documentation

8.6.3.1 operator=()

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

8.6.4 Friends And Related Function Documentation

8.6.4.1 init_attrjeod__MassPointLinks

```
void init_attrjeod__MassPointLinks ( )  [friend]
```

8.6.4.2 InputProcessor

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

Definition at line 83 of file mass_point_links.hh.

8.6.5 Field Documentation

8.6.5.1 default_path_size

```
const unsigned int jeod::MassPointLinks::default_path_size = 8  [static], [private]
```

Definition at line 106 of file mass_point_links.hh.

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

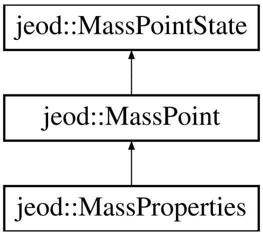
- mass_point_links.hh

8.7 jeod::MassPointState Class Reference

Defines the state – position and orientation – of a MassPoint.

```
#include <mass_point_state.hh>
```

Inheritance diagram for jeod::MassPointState:



Public Member Functions

- MassPointState ()=default
- virtual ∼MassPointState ()=default
- virtual void initialize_mass_point ()

    Initialize a mass point.
- void update_point (const double pt_location[3])

    Update the mass point's structure-to-point vector.
- void update_orientation (const double transformation[3][3])

    Update the mass point's structure-to-point frame transformation matrix and items derived from it.
- void update_orientation (const Quaternion &left_quat)

    Update the mass point's structure-to-point frame transformation matrix and items derived from it.
- void compute_quaternion ()

    Compute quaternion.
- void compute_transformation ()

    Compute matrix.
- void copy_state (const MassPointState &source)

    Copy the provided state to this.
- void negate (const MassPointState &source)

    Copy a mass point state, negated.
- void incr_left (const MassPointState &s_ab)

    Compute S_A:C = S_A:B + S_B:C, with this initially containing S_B:C, the supplied argument containing S_A:B, and the resultant composition of states stored in this.
- void incr_right (const MassPointState &s_bc)

    Compute S_A:C = S_A:B + S_B:C, with this initially containing S_A:B, the supplied argument containing S_B:C, and the resultant composition of states stored in this.
- void decr_left (const MassPointState &s_ab)

    Compute S_B:C = (-S_A:B) + S_A:C, with this initially containing S_A:C, the supplied argument containing S_A:B, and the resultant composition of states stored in this.
- void decr_right (const MassPointState &s_bc)

    Compute S_A:B = S_A:C + (-S_B:C) with this initially containing S_A:C, the supplied argument containing S_B:C, and the resultant composition of states stored in this.

Data Fields

- double position [3] {}

    Mass point location with respect to the origin of some parent frame and expressed in the parent frame's coordinates.
- Quaternion Q_parent_this

    Left transformation quaternion from the parent frame to the frame associated with the mass point.
- double T_parent_this [3][3] {IDENTITY_3X3}

    transformation matrix from the parent frame to the frame associated with the mass point.

Friends

- class InputProcessor
- void init_attrjeod__MassPointState ()

8.7.1 Detailed Description

Defines the state – position and orientation – of a MassPoint.

Definition at line 84 of file mass_point_state.hh.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 MassPointState()

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

8.7.2.2 ∼MassPointState()

```
virtual jeod::MassPointState::∼MassPointState ( )  [virtual], [default]
```

8.7.3 Member Function Documentation

8.7.3.1 compute_quaternion()

```
void jeod::MassPointState::compute_quaternion ( )  [inline]
```

Compute quaternion.

Someone changed the matrix under our noses.

Definition at line 191 of file mass_point_state.hh.

8.7.3.2 compute_transformation()

```
void jeod::MassPointState::compute_transformation ( )  [inline]
```

Compute matrix.

Someone changed the quaternion under our noses.

Definition at line 199 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_update_properties(), decr_left(), decr_right(), incr_left(), incr_right(), and jeod::MassBody::reattach().

8.7.3.3 copy_state()

```
void jeod::MassPointState::copy_state (
            const MassPointState & source )  [inline]
```

Copy the provided state to this.

Parameters

in	*source*	Source state

Definition at line 209 of file mass_point_state.hh.

8.7.3.4 decr_left()

```
void jeod::MassPointState::decr_left (
            const MassPointState & s_ab )
```

Compute S_B:C = (-S_A:B) + S_A:C, with this initially containing S_A:C, the supplied argument containing S_A:B, and the resultant composition of states stored in this.

Parameters

in	*s_ab*	Left addend

Definition at line 240 of file mass_point_state.cc.

References compute_transformation(), position, Q_parent_this, and T_parent_this.

Referenced by jeod::MassPoint::compute_relative_state().

8.7.3.5 decr_right()

```
void jeod::MassPointState::decr_right (
            const MassPointState & s_bc )
```

Compute S_A:B = S_A:C + (-S_B:C) with this initially containing S_A:C, the supplied argument containing S_B:C, and the resultant composition of states stored in this.

Parameters

in	*s_bc*	Left addend

Definition at line 283 of file mass_point_state.cc.

References compute_transformation(), position, Q_parent_this, and T_parent_this.

8.7.3.6 incr_left()

```
void jeod::MassPointState::incr_left (
            const MassPointState & s_ab )
```

Compute S_A:C = S_A:B + S_B:C, with this initially containing S_B:C, the supplied argument containing S_A:B, and the resultant composition of states stored in this.

Parameters

in	*s_ab*	Left addend

Definition at line 154 of file mass_point_state.cc.

References compute_transformation(), position, Q_parent_this, and T_parent_this.

8.7.3.7 incr_right()

```
void jeod::MassPointState::incr_right (
             const MassPointState & s_bc )
```

Compute S_A:C = S_A:B + S_B:C, with this initially containing S_A:B, the supplied argument containing S_B:C, and the resultant composition of states stored in this.

Parameters

in	*s_bc*	Right addend

Definition at line 195 of file mass_point_state.cc.

References compute_transformation(), position, Q_parent_this, and T_parent_this.

8.7.3.8 initialize_mass_point()

```
void jeod::MassPointState::initialize_mass_point ( )  [virtual]
```

Initialize a mass point.

Reimplemented in jeod::MassPoint.

Definition at line 52 of file mass_point_state.cc.

References position, Q_parent_this, and T_parent_this.

Referenced by jeod::MassPoint::compute_pred_rel_state(), jeod::MassPoint::compute_relative_state(), jeod::MassPoint::compute_state_wrt_pred(), and jeod::MassPoint::initialize_mass_point().

8.7.3.9 negate()

```
void jeod::MassPointState::negate (
             const MassPointState & source )
```

Copy a mass point state, negated.

Parameters

in	*source*	Source state

Definition at line 120 of file mass_point_state.cc.

References position, Q_parent_this, and T_parent_this.

8.7.3.10 update_orientation() [1/2]

```
void jeod::MassPointState::update_orientation (
            const double transformation[3][3] )  [inline]
```

Update the mass point's structure-to-point frame transformation matrix and items derived from it.

Parameters

in	*transformation*	New structure-to-point xform

Definition at line 169 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_update_properties(), and jeod::MassBody::reattach().

8.7.3.11 update_orientation() [2/2]

```
void jeod::MassPointState::update_orientation (
            const Quaternion & left_quat )  [inline]
```

Update the mass point's structure-to-point frame transformation matrix and items derived from it.

Parameters

in	*left_quat*	New structure-to-point quat

Definition at line 181 of file mass_point_state.hh.

8.7.3.12 update_point()

```
void jeod::MassPointState::update_point (
            const double pt_location[3] )  [inline]
```

Update the mass point's structure-to-point vector.

Parameters

in	*pt_location*	New location, struct coords. Units: M

Definition at line 158 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_update_properties(), jeod::MassPointInit::initialize_mass_point(), and jeod::MassBody::reattach().

8.7.4 Friends And Related Function Documentation

8.7.4.1 init_attrjeod__MassPointState

```
void init_attrjeod__MassPointState ( )  [friend]
```

8.7.4.2 InputProcessor

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

Definition at line 86 of file mass_point_state.hh.

8.7.5 Field Documentation

8.7.5.1 position

```
double jeod::MassPointState::position[3] {}
```

Mass point location with respect to the origin of some parent frame and expressed in the parent frame's coordinates.

trick_units(m)

Definition at line 92 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_to(), jeod::MassBody::attach_update_properties(), jeod::MassBody::calc_composite_cm(), jeod::MassBody::calc_composite_inertia(), jeod::MassPoint::compute_pred_rel_state(), jeod::MassPoint::compute_state_wrt_pred(), decr_left(), decr_right(), incr_left(), incr_right(), initialize_mass_point(), jeod::MassPropertiesInit::initialize_mass_properties(), negate(), jeod::MassBody::print_body(), jeod::MassBody::reattach(), and jeod::MassBody::update_mass_properties().

8.7.5.2 Q_parent_this

```
Quaternion jeod::MassPointState::Q_parent_this
```

Left transformation quaternion from the parent frame to the frame associated with the mass point.

trick_units(–)

Definition at line 98 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_to(), jeod::MassBody::attach_update_properties(), jeod::MassPoint↩ ::compute_pred_rel_state(), jeod::MassPoint::compute_state_wrt_pred(), decr_left(), decr_right(), incr_left(), incr↩ _right(), jeod::MassBody::initialize_mass(), initialize_mass_point(), jeod::MassPointInit::initialize_mass_point(), negate(), and jeod::MassBody::reattach().

8.7.5.3 T_parent_this

```
double jeod::MassPointState::T_parent_this[3][3] {IDENTITY_3X3}
```

transformation matrix from the parent frame to the frame associated with the mass point.

NOTE: The MassPointState infrastructure ensures that the quaternion and matrix representations are equivalent.↩ trick_units(–)

Definition at line 106 of file mass_point_state.hh.

Referenced by jeod::MassBody::attach_to(), jeod::MassBody::attach_update_properties(), jeod::MassBody::calc↩ _composite_inertia(), jeod::MassPoint::compute_pred_rel_state(), jeod::MassPoint::compute_state_wrt_pred(), decr_left(), decr_right(), incr_left(), incr_right(), jeod::MassBody::initialize_mass(), initialize_mass_point(), jeod↩ ::MassPointInit::initialize_mass_point(), jeod::MassPropertiesInit::initialize_mass_properties(), jeod::MassBody::↩ MassBody(), negate(), jeod::MassBody::print_body(), jeod::MassBody::reattach(), and jeod::MassBody::update_↩ mass_properties().

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

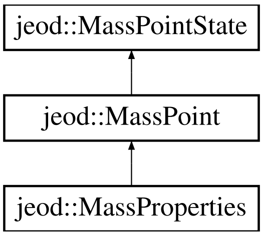
- mass_point_state.hh
- mass_point_state.cc

8.8 jeod::MassProperties Class Reference

Defines mass properties – mass and inertia tensor.

```
#include <mass_properties.hh>
```

Inheritance diagram for jeod::MassProperties:



Public Member Functions

- MassProperties ()=default
- MassProperties (const MassProperties &)=delete
- MassProperties & operator= (const MassProperties &)=delete

Data Fields

- double mass {}

  Mass of the subject mass element (always >= 0).
- double inertia [3][3] {}

  Inertia tensor of the subject mass element about the subject mass element's center of mass, expressed in subject mass element body coordinates.
- double inverse_mass {}

  The inverse of mass.
- double inverse_inertia [3][3] {}

  The inverse of the composite inertia.

Friends

- class InputProcessor
- void init_attrjeod__MassProperties ()

Additional Inherited Members

8.8.1 Detailed Description

Defines mass properties – mass and inertia tensor.

A MassBody contains two MassProperties objects that represent the body's core and composite mass properties. A MassProperties object is-a MassPoint and hence a MassPointState by inheritance. The parent point of a MassProperties object is always the structural frame of the MassBody that 'owns' the MassProperties object.

Note that a MassProperties does not explicitly define the center of mass location and body frame orientation. These properties are defined by elements inherited from MassPointState:

- position defines the center of mass in the structural frame.
- Q_parent_this/T_parent_this define the orientation of the body frame with respect to the structural frame.

Definition at line 86 of file mass_properties.hh.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 MassProperties() [1/2]

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

8.8.2.2 MassProperties() [2/2]

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

8.8.3 Member Function Documentation

8.8.3.1 operator=()

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

8.8.4 Friends And Related Function Documentation

8.8.4.1 init_attrjeod__MassProperties

```
void init_attrjeod__MassProperties ( )  [friend]
```

8.8.4.2 InputProcessor

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

Definition at line 88 of file mass_properties.hh.

8.8.5 Field Documentation

8.8.5.1 inertia

```
double jeod::MassProperties::inertia[3][3] {}
```

Inertia tensor of the subject mass element about the subject mass element's center of mass, expressed in subject mass element body coordinates.

Note: The diagonal elements of the inertia tensor are positive moments of inertial while the off-diagonal elements are megative products of inertia.trick_units(kg∗m2)

Definition at line 104 of file mass_properties.hh.

Referenced by jeod::MassBody::calc_composite_inertia(), jeod::MassBody::detach_update_properties(), jeod↩ ::MassPropertiesInit::initialize_mass_properties(), jeod::MassBody::print_body(), and jeod::MassBody::update_↩ mass_properties().

8.8.5.2 inverse_inertia

```
double jeod::MassProperties::inverse_inertia[3][3] {}
```

The inverse of the composite inertia.

Used only for root dynamic bodies composite props.trick_units(1/kg/m2)

Definition at line 116 of file mass_properties.hh.

Referenced by jeod::MassBody::detach_update_properties(), jeod::MassBody::print_body(), and jeod::Mass↩ Body::update_mass_properties().

8.8.5.3 inverse_mass

```
double jeod::MassProperties::inverse_mass {}
```

The inverse of mass.

Used only for root dynamic bodies composite props.trick_units(1/kg)

Definition at line 110 of file mass_properties.hh.

Referenced by jeod::MassBody::calc_composite_cm(), jeod::MassBody::print_body(), and jeod::MassBody↩ ::update_mass_properties().

8.8.5.4 mass

```
double jeod::MassProperties::mass {}
```

Mass of the subject mass element (always >= 0).

The core mass element's mass is set externally while the composite mass element's mass is computed by the mass model.trick_units(kg)

Definition at line 95 of file mass_properties.hh.

Referenced by jeod::MassBody::calc_composite_cm(), jeod::MassBody::calc_composite_inertia(), jeod::MassBody::detach_update_properties(), jeod::MassPropertiesInit::initialize_mass_properties(), jeod::MassBody::print_body(), and jeod::MassBody::update_mass_properties().

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

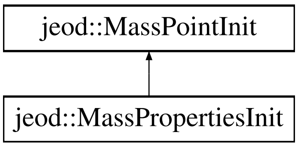
- mass_properties.hh

8.9 jeod::MassPropertiesInit Class Reference

Contains data used to initialize a mass model object.

```
#include <mass_properties_init.hh>
```

Inheritance diagram for jeod::MassPropertiesInit:



Public Types

- enum InertiaSpec {
NoSpec = 0, Body = 1, StructCG = 2, Struct = 3,
SpecCG = 4, Spec = 5 }

  Specifies the origin and axes of the input inertia tensor.

Public Member Functions

- MassPropertiesInit ()

  Default constructor; constructs a MassPropertiesInit object.
- void initialize_mass_properties (MassProperties &mass_properties) const

  Initialize the subject MassProperties.

Data Fields

- double mass {}

  Mass of the core body.
- double ∗ cm

  Location of the core body center of mass in the structural frame.
- double inertia [3][3] {}

  Inertia tensor of the core body in an arbitrary reference frame.
- InertiaSpec inertia_spec {Body}

  Indicates how the user has specified the core inertia matrix.
- double inertia_offset [3] {}

  Offset from inertia reference frame to the core center of mass, expressed in inertia reference frame coordinates.
- Orientation inertia_orientation

  Orientation of body frame wrt inertia reference frame, i.e.

Friends

- class InputProcessor
- void init_attrjeod__MassPropertiesInit ()

8.9.1 Detailed Description

Contains data used to initialize a mass model object.

Definition at line 79 of file mass_properties_init.hh.

8.9.2 Member Enumeration Documentation

8.9.2.1 InertiaSpec

```
enum jeod::MassPropertiesInit::InertiaSpec
```

Specifies the origin and axes of the input inertia tensor.

Enumerator

NoSpec	Inertia matrix is specified directly.
Body	Initial inertia frame is body frame.
StructCG	Initial inertia frame is struct at CG.
Struct	Initial inertia frame is structural frame.
SpecCG	Initial inertia frame is in frame at CG.
Spec	Initial inertia frame is specified frame.

Definition at line 87 of file mass_properties_init.hh.

8.9.3 Constructor & Destructor Documentation

8.9.3.1 MassPropertiesInit()

```
jeod::MassPropertiesInit::MassPropertiesInit ( )
```

Default constructor; constructs a MassPropertiesInit object.

Definition at line 55 of file mass_properties_init.cc.

References cm, and jeod::MassPointInit::position.

8.9.4 Member Function Documentation

8.9.4.1 initialize_mass_properties()

```
void jeod::MassPropertiesInit::initialize_mass_properties (
             MassProperties & properties ) const
```

Initialize the subject MassProperties.

Parameters

out	*properties*	Properties to initialize

Definition at line 65 of file mass_properties_init.cc.

References Body, jeod::MassBody::compute_point_mass_inertia(), jeod::MassProperties::inertia, inertia, inertia↩_offset, inertia_orientation, inertia_spec, jeod::MassPointInit::initialize_mass_point(), jeod::MassBodyMessages↩::invalid_enum, jeod::MassProperties::mass, mass, NoSpec, jeod::MassPointState::position, Spec, SpecCG, Struct, StructCG, and jeod::MassPointState::T_parent_this.

Referenced by jeod::MassBody::initialize_mass().

8.9.5 Friends And Related Function Documentation

8.9.5.1 init_attrjeod__MassPropertiesInit

```
void init_attrjeod__MassPropertiesInit ( )  [friend]
```

8.9.5.2 InputProcessor

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

Definition at line 81 of file mass_properties_init.hh.

8.9.6 Field Documentation

8.9.6.1 cm

```
double* jeod::MassPropertiesInit::cm
```

Location of the core body center of mass in the structural frame.

This is just an alias for the generic position member.trick_units(m)

Definition at line 108 of file mass_properties_init.hh.

Referenced by MassPropertiesInit().

8.9.6.2 inertia

```
double jeod::MassPropertiesInit::inertia[3][3] {}
```

Inertia tensor of the core body in an arbitrary reference frame.

trick_units(kg∗m2)

Definition at line 113 of file mass_properties_init.hh.

Referenced by initialize_mass_properties().

8.9.6.3 inertia_offset

```
double jeod::MassPropertiesInit::inertia_offset[3] {}
```

Offset from inertia reference frame to the core center of mass, expressed in inertia reference frame coordinates.

trick_units(m)

Definition at line 124 of file mass_properties_init.hh.

Referenced by initialize_mass_properties().

8.9.6.4 inertia_orientation

```
Orientation jeod::MassPropertiesInit::inertia_orientation
```

Orientation of body frame wrt inertia reference frame, i.e.

the process by which the frame in which the inertia tensor is specified may be transformed to the body frame, e.g. T_{spec->body}.trick_units(–)

Definition at line 131 of file mass_properties_init.hh.

Referenced by initialize_mass_properties().

8.9.6.5 inertia_spec

```
InertiaSpec jeod::MassPropertiesInit::inertia_spec {Body}
```

Indicates how the user has specified the core inertia matrix.

trick_units(–)

Definition at line 118 of file mass_properties_init.hh.

Referenced by initialize_mass_properties().

8.9.6.6 mass

```
double jeod::MassPropertiesInit::mass {}
```

Mass of the core body.

trick_units(kg)

Definition at line 102 of file mass_properties_init.hh.

Referenced by initialize_mass_properties().

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

- mass_properties_init.hh
- mass_properties_init.cc

Chapter 9

File Documentation

9.1 class_declarations.hh File Reference

Forward declarations of mass model classes.

Namespaces

- jeod

  Namespace jeod.

9.1.1 Detailed Description

Forward declarations of mass model classes.

9.2 mass.cc File Reference

Define methods for the MassBody class.

```
#include <cstddef>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/dyn_manager/include/dyn_manager.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/named_item/include/named_item.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
#include "../include/mass_properties_init.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.2.1 Detailed Description

Define methods for the MassBody class.

9.3 mass.hh File Reference

Define the class MassBody.

```
#include <list>
#include "class_declarations.hh"
#include "mass_body_links.hh"
#include "mass_point.hh"
#include "mass_point_init.hh"
#include "mass_properties.hh"
#include "dynamics/dyn_body/include/class_declarations.hh"
#include "utils/container/include/pointer_list.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <cstddef>
#include <cstdio>
#include <string>
#include <utility>
```

Data Structures

- class jeod::MassBody

    Represent both an atomic chunk of mass and an interconnected set of masses.

Namespaces

- jeod

    Namespace jeod.

9.3.1 Detailed Description

Define the class MassBody.

A MassBody represents both some atomic chunk of mass and an interconnected set of masses, and contains mass properties to represent both this atomic chunk of mass and the composite chunks of mass connected to this MassBody.

9.4 mass_attach.cc File Reference

Define MassBody attach methods.

```
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
#include "dynamics/dyn_manager/include/dyn_manager.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include <cstddef>
```

Namespaces

- jeod

  Namespace jeod.

Functions

- static void jeod::generate_bad_point_message (const char ∗file, unsigned int line, const std::string &child_body_name, const std::string &child_point_name, const MassPoint ∗child_point, const std::string &parent_body_name, const std::string &parent_point_name, const MassPoint ∗parent_point)

  Generate a message regarding failure to find mass points on a vehicle.

9.4.1 Detailed Description

Define MassBody attach methods.

9.5 mass_body_links.hh File Reference

Define the class MassBodyLinks, the class that encapsulates the links between mass bodies.

```
#include "utils/ref_frames/include/tree_links.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "class_declarations.hh"
#include "mass_messages.hh"
```

Data Structures

- class jeod::MassBodyLinks

  Encapsulates the links between mass bodies.

Namespaces

- jeod

  Namespace jeod.

9.5.1 Detailed Description

Define the class MassBodyLinks, the class that encapsulates the links between mass bodies.

9.6 mass_calc_composite_cm.cc File Reference

Calculate the aggregate cm for a composite MassBody.

```
#include "utils/math/include/vector3.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/mass.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.6.1 Detailed Description

Calculate the aggregate cm for a composite MassBody.

9.7 mass_calc_composite_inertia.cc File Reference

Calculate the aggregate inertia matrix for a composite MassBody.

```
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/mass.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.7.1 Detailed Description

Calculate the aggregate inertia matrix for a composite MassBody.

9.8 mass_detach.cc File Reference

Define MassBody detachment methods.

```
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/dyn_manager/include/dyn_manager.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include <cstddef>
```

Namespaces

- jeod

  Namespace jeod.

9.8.1 Detailed Description

Define MassBody detachment methods.

9.9 mass_messages.cc File Reference

Implement the class MassBodyMessages.

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

Namespaces

- jeod

  Namespace jeod.

Macros

- #define MAKE_MASSBODY_MESSAGE_CODE(id) JEOD_MAKE_MESSAGE_CODE(MassBody↩ Messages, "dynamics/mass/", id)

9.9.1 Detailed Description

Implement the class MassBodyMessages.

9.9.2 Macro Definition Documentation

9.9.2.1 MAKE_MASSBODY_MESSAGE_CODE

```
#define MAKE_MASSBODY_MESSAGE_CODE(
              id ) JEOD_MAKE_MESSAGE_CODE(MassBodyMessages, "dynamics/mass/", id)
```

Definition at line 43 of file mass_messages.cc.

9.10 mass_messages.hh File Reference

Define the class MassBodyMessages, the class that specifies the message IDs used in the MassBody model.

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

Data Structures

- class jeod::MassBodyMessages

  Specify the message IDs used in the MassBody model.

Namespaces

- jeod

  Namespace jeod.

9.10.1 Detailed Description

Define the class MassBodyMessages, the class that specifies the message IDs used in the MassBody model.

9.11 mass_point.cc File Reference

Define basic methods for the MassPoint class.

```
#include "../include/mass_point.hh"
#include "../include/mass_messages.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include <cstddef>
```

Namespaces

- jeod

  Namespace jeod.

9.11.1 Detailed Description

Define basic methods for the MassPoint class.

9.12 mass_point.hh File Reference

Define the class MassPoint, which defines the base features of a point related to a MassBody.

```
#include "class_declarations.hh"
#include "mass_point_links.hh"
#include "mass_point_state.hh"
#include "utils/named_item/include/named_item.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <string>
#include <utility>
```

Data Structures

- class jeod::MassPoint

  Adds tree linkages and a name to a MassPointState.

Namespaces

- jeod

  Namespace jeod.

9.12.1 Detailed Description

Define the class MassPoint, which defines the base features of a point related to a MassBody.

A MassPoint is simply a MassPointState with a name plus a pointer to a "parent" MassPoint.

9.13 mass_point_init.cc File Reference

Define methods for the MassPointInit class.

```
#include "../include/mass_point_init.hh"
#include "../include/mass_messages.hh"
#include "../include/mass_point.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.13.1 Detailed Description

Define methods for the MassPointInit class.

9.14 mass_point_init.hh File Reference

Define the class MassPointInit, which initializes a MassPoint object.

```
#include "class_declarations.hh"
#include "utils/orientation/include/orientation.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <string>
#include <utility>
```

Data Structures

- class jeod::MassPointInit

  Contains data used to initialize a MassPoint.

Namespaces

- jeod

  Namespace jeod.

9.14.1 Detailed Description

Define the class MassPointInit, which initializes a MassPoint object.

9.15 mass_point_links.hh File Reference

Define the class MassPointLinks, the class that encapsulates the links between mass points.

```
#include "class_declarations.hh"
#include "mass_messages.hh"
#include "utils/ref_frames/include/tree_links.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class jeod::MassPointLinks

  Encapsulates the links between mass points.

Namespaces

- jeod

  Namespace jeod.

9.15.1 Detailed Description

Define the class MassPointLinks, the class that encapsulates the links between mass points.

9.16 mass_point_mass_inertia.cc File Reference

Compute the inertia tensor of a point mass.

```
#include "../include/mass.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.16.1 Detailed Description

Compute the inertia tensor of a point mass.

9.17 mass_point_state.cc File Reference

Define basic methods for the MassPointState class.

```
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/numerical.hh"
#include "utils/math/include/vector3.hh"
#include "../include/mass_point_state.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.17.1 Detailed Description

Define basic methods for the MassPointState class.

9.18 mass_point_state.hh File Reference

Define the class MassPointState, which defines the state – position and orientation – of a MassPoint.

```
#include "class_declarations.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/quaternion/include/quat.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/math/include/macro_def.hh"
#include "utils/math/include/macro_undef.hh"
```

Data Structures

- class jeod::MassPointState

    Defines the state – position and orientation – of a MassPoint.

Namespaces

- jeod

    Namespace jeod.

9.18.1 Detailed Description

Define the class MassPointState, which defines the state – position and orientation – of a MassPoint.

9.19 mass_print_body.cc File Reference

Define MassBody::print_body.

```
#include <cstdio>
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/mass.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.19.1 Detailed Description

Define MassBody::print_body.

9.20 mass_print_tree.cc File Reference

Print out the MassTree members' data to a given file.

```
#include <cstddef>
#include <cstdio>
#include "utils/message/include/message_handler.hh"
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.20.1 Detailed Description

Print out the MassTree members' data to a given file.

9.21 mass_properties.hh File Reference

Define the class MassProperties.

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

Data Structures

- class jeod::MassProperties

  Defines mass properties – mass and inertia tensor.

Namespaces

- jeod

  Namespace jeod.

9.21.1 Detailed Description

Define the class MassProperties.

9.22 mass_properties_init.cc File Reference

Define methods for the MassPropertiesInit class.

```
#include "../include/mass_properties_init.hh"
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/message/include/message_handler.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.22.1 Detailed Description

Define methods for the MassPropertiesInit class.

9.23 mass_properties_init.hh File Reference

Define class MassPropertiesInit, which initializes a mass properties object.

```
#include "class_declarations.hh"
#include "mass_point_init.hh"
#include "utils/orientation/include/orientation.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

- class jeod::MassPropertiesInit

  Contains data used to initialize a mass model object.

Namespaces

- jeod

  Namespace jeod.

9.23.1 Detailed Description

Define class MassPropertiesInit, which initializes a mass properties object.

9.24 mass_reattach.cc File Reference

Define MassBody::reattach.

```
#include "utils/message/include/message_handler.hh"
#include "../include/mass.hh"
#include "../include/mass_messages.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.24.1 Detailed Description

Define MassBody::reattach.

9.25 mass_update.cc File Reference

Define MassBody::update_mass_properties.

```
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/mass.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.25.1 Detailed Description

Define MassBody::update_mass_properties.

Index