AerodynamicsModel

5.1

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

- Interactions

6.1.1 Detailed Description

6.2 Interactions

Modules

- Aerodynamics

6.2.1 Detailed Description

6.3 Aerodynamics

Files

- file aero_drag.hh

  Orbital aerodynamics paramter definitions and the main class for calculating aerodynamic drag.
- file aero_facet.hh

  Individual facets for use with aero environment interaction models.
- file aero_params.hh

  A virtual base class for aero facet parameters, used to create interaction facets for aero in the InteractionSurface←↩ Factorys.
- file aero_surface.hh

  Vehicle surface model for aerodynamics.
- file aero_surface_factory.hh

  Factory that creates an aerodynamic interaction surface from a surface model.
- file aerodynamics_messages.hh

  Aerodynamics message for message handling.
- file class_declarations.hh

  Forward declaration of classes defined in the aerodynamics package.
- file default_aero.hh

  An implementation of ballistic coefficent and coefficient of drag for use in the AerodynamicDrag object.
- file flat_plate_aero_facet.hh

  The aerodynamic specific implementation of flat plate.
- file flat_plate_aero_factory.hh

  Creates a flat plate aero facet from a basic flat plate facet.
- file flat_plate_aero_params.hh

  The set of parameters used to create an aerodynamic interaction specific flat plate facet from a general flat plate facet.
- file flat_plate_thermal_aero_factory.hh

  Creates a flat plate aero facet from a flat plate thermal facet.
- file aero_drag.cc

  Orbital aerodynamic force and torque computation, and related classes.
- file aero_surface.cc

  Vehicle surface model for the aerodyanmic interaction models.
- file aero_surface_factory.cc

  Factory that creates an aerodynamics surface, from a surface model.
- file aerodynamics_messages.cc

  Implement aerodynamics_messages.
- file default_aero.cc

  Implement a virtual base class for aerodynamic drag calculations.
- file flat_plate_aero_facet.cc

  Individual facets for use with aero environment interaction models.
- file flat_plate_aero_factory.cc

  Factory that creates a FlatPlateAeroFacet from a FlatPlate, using a FlatPlateAeroParams object.
- file flat_plate_thermal_aero_factory.cc

  Factory that creates a FlatPlateAeroFacet from a FlatPlateThermal, using a FlatPlateAeroParams object.

Namespaces

- jeod

  Namespace jeod.

Macros

- #define PATH "interactions/aerodynamics/"

6.3.1 Detailed Description

6.3.2 Macro Definition Documentation

6.3.2.1 PATH

```
#define PATH "interactions/aerodynamics/"
```

Definition at line 36 of file aerodynamics_messages.cc.

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class AeroDragEnum

  Contains enumerations associated with aerodynamic drag.
- class AeroDragParameters

  Contains parameters associated with aerodynamic drag.
- class AerodynamicDrag

  The main class for calculating aerodynamic drag.
- class AerodynamicDrag_aero_model_default_data
- class AerodynamicsMessages

  Messages associated with use of the aerodynamics model.
- class AeroFacet

  An aerodynamic interaction specific facet for use in the surface model.
- class AeroParams

  A base class for all aerodynamic parameters used in the surface model.
- class AeroSurface

  The aerodynamic specific interaction surface, for use with the surface model.
- class AeroSurfaceFactory

  The surface factory that creates an aerodynamic specific surface from a general surface.
- class DefaultAero

  The simple, default, aerodynamic drag model, including coefficient of drag, ballistic coefficient, etc.
- class FlatPlateAeroFacet

  The aerdynamic specific version of a flat plate.
- class FlatPlateAeroFactory

  Creates a FlatPlateAeroFacet from a FlatPlate.
- class FlatPlateAeroParams

  used in the FlatPlateAeroFactory to create a FlatPlateAeroFacet.
- class FlatPlateThermalAeroFactory

  Creates a FlatPlateAeroFacet from a FlatPlate.

7.1.1 Detailed Description

Namespace jeod.

Chapter 8

Data Structure Documentation

8.1 jeod::AeroDragEnum Class Reference

Contains enumerations associated with aerodynamic drag.

```
#include <aero_drag.hh>
```

Public Types

- enum CoefCalcMethod { Specular = 0, Diffuse, Mixed, Calc_coef }
  Dictates how the coefficients of drag will be calculated when using a flat plate model.

8.1.1 Detailed Description

Contains enumerations associated with aerodynamic drag.

Definition at line 87 of file aero_drag.hh.

8.1.2 Member Enumeration Documentation

8.1.2.1 CoefCalcMethod

```
enum jeod::AeroDragEnum::CoefCalcMethod
```

Dictates how the coefficients of drag will be calculated when using a flat plate model.

Enumerator

Specular	
Diffuse	
Mixed	
Calc_coef	

Definition at line 94 of file aero_drag.hh.

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

- aero_drag.hh

8.2 jeod::AeroDragParameters Class Reference

Contains parameters associated with aerodynamic drag.

```
#include <aero_drag.hh>
```

Data Fields

- double dynamic_pressure {}

    dynamic pressure, 0.5 ∗ density ∗ velocity^2
- double gas_const {}

    gas constant, ala PV = mRT; R = 287 for air.
- double temp_free_stream {}

    temperature of the incident stream of free molecular flow.

8.2.1 Detailed Description

Contains parameters associated with aerodynamic drag.

Definition at line 114 of file aero_drag.hh.

8.2.2 Field Documentation

8.2.2.1 dynamic_pressure

```
double jeod::AeroDragParameters::dynamic_pressure {}
```

dynamic pressure, 0.5 ∗ density ∗ velocity^2

trick_units(N/m2)

Definition at line 120 of file aero_drag.hh.

Referenced by jeod::AerodynamicDrag::aero_drag(), jeod::FlatPlateAeroFacet::aerodrag_force(), and jeod::DefaultAero::aerodrag_force().

8.2.2.2 gas_const

```
double jeod::AeroDragParameters::gas_const {}
```

gas constant, ala PV = mRT; R = 287 for air.

trick_units(N∗m/kg/K)

Definition at line 125 of file aero_drag.hh.

Referenced by jeod::FlatPlateAeroFacet::aerodrag_force(), and jeod::AerodynamicDrag_aero_model_default_data::initialize().

8.2.2.3 temp_free_stream

```
double jeod::AeroDragParameters::temp_free_stream {}
```

temperature of the incident stream of free molecular flow.

trick_units(K)

Definition at line 130 of file aero_drag.hh.

Referenced by jeod::FlatPlateAeroFacet::aerodrag_force(), and jeod::AerodynamicDrag_aero_model_default_data::initialize().

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

- aero_drag.hh

8.3 jeod::AerodynamicDrag Class Reference

The main class for calculating aerodynamic drag.

```
#include <aero_drag.hh>
```

Public Member Functions

- AerodynamicDrag ()

  default constructor Return: – void
- virtual ∼AerodynamicDrag ()=default
- AerodynamicDrag (const AerodynamicDrag &)=delete
- AerodynamicDrag & operator= (const AerodynamicDrag &)=delete
- void aero_drag (double inertial_velocity[3], AtmosphereState ∗atmos_ptr, double T_inertial_struct[3][3], double mass, double center_grav[3])

  Calculates the total aerodynamic drag force and torque, from the information given.
- void set_aero_surface (AeroSurface &to_set)

  Set the surface this AeroDrag object will calculate drag for.
- void clear_aero_surface ()

  Remove any AeroSurface being used for calculation.

Data Fields

- bool active {true}

  On = aerodynamics enabled.
- bool constant_density {}

  Use constant density for aero drag?
- double density {}

  Density of the last time AerodynamicDrag was used.
- double aero_force [3] {}

  Total Force due to aero drag, resulting from all plates combined.
- double aero_torque [3] {}

  Total torque due to aero drag, resulting from all plates combined.
- AeroDragParameters param

  parameters shared with plate model
- bool use_default_behavior {true}

  Use the default behavior?
- AeroSurface ∗ aero_surface_ptr {}

  Pointer to the current aero surface.
- DefaultAero ∗ default_behavior

  Pointer to an object that defines the default aero behavior.
- DefaultAero ballistic_drag

  Spherical, ballistic drag.

8.3.1 Detailed Description

The main class for calculating aerodynamic drag.

Definition at line 136 of file aero_drag.hh.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 AerodynamicDrag() [1/2]

```
jeod::AerodynamicDrag::AerodynamicDrag ( )
```

default constructor Return: – void

Definition at line 68 of file aero_drag.cc.

8.3.2.2 ∼AerodynamicDrag()

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

8.3.2.3 AerodynamicDrag() [2/2]

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

8.3.3 Member Function Documentation

8.3.3.1 aero_drag()

```
void jeod::AerodynamicDrag::aero_drag (
            double inertial_velocity[3],
            AtmosphereState * atmos_ptr,
            double T_inertial_struct[3][3],
            double mass,
            double center_grav[3] )
```

Calculates the total aerodynamic drag force and torque, from the information given.

Parameters

in	*inertial_velocity*	vehicle velocity in inertial RF Units: M/s
in	*atmos_ptr*	Pointer to the AtmosphereState used for density and wind information
in	*T_inertial_struct*	Transformation matrix from the inertial frame to the structural
in	*mass*	kg Mass of the vehicle
in	*center_grav*	position of the center of gravity, in the structural frame Units: M

Definition at line 83 of file aero_drag.cc.

References active, jeod::AeroSurface::aero_facets, aero_force, aero_surface_ptr, aero_torque, jeod::AeroFacet::aerodrag_force(), jeod::DefaultAero::aerodrag_force(), constant_density, default_behavior, density, jeod::AeroDragParameters::dynamic_pressure, jeod::AeroSurface::facets_size, param, jeod::AerodynamicsMessages::runtime_error, and use_default_behavior.

8.3.3.2 clear_aero_surface()

```
void jeod::AerodynamicDrag::clear_aero_surface ( )
```

Remove any AeroSurface being used for calculation.

Note: The variable "use_default_behavior" must be set to true if there is no set aero surface

Definition at line 182 of file aero_drag.cc.

References aero_surface_ptr.

8.3.3.3 operator=()

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

8.3.3.4 set_aero_surface()

```
void jeod::AerodynamicDrag::set_aero_surface (
             AeroSurface & to_set )
```

Set the surface this AeroDrag object will calculate drag for.

Parameters

in	*to_set*	The AeroSurface to be used

Definition at line 172 of file aero_drag.cc.

References aero_surface_ptr.

8.3.4 Field Documentation

8.3.4.1 active

```
bool jeod::AerodynamicDrag::active {true}
```

On = aerodynamics enabled.

trick_units(--)

Definition at line 142 of file aero_drag.hh.

Referenced by aero_drag(), and jeod::AerodynamicDrag_aero_model_default_data::initialize().

8.3.4.2 aero_force

```
double jeod::AerodynamicDrag::aero_force[3] {}
```

Total Force due to aero drag, resulting from all plates combined.

trick_units(N)

Definition at line 159 of file aero_drag.hh.

Referenced by aero_drag().

8.3.4.3 aero_surface_ptr

```
AeroSurface* jeod::AerodynamicDrag::aero_surface_ptr {}
```

Pointer to the current aero surface.

trick_units(–)

Definition at line 179 of file aero_drag.hh.

Referenced by aero_drag(), clear_aero_surface(), and set_aero_surface().

8.3.4.4 aero_torque

```
double jeod::AerodynamicDrag::aero_torque[3] {}
```

Total torque due to aero drag, resulting from all plates combined.

trick_units(N∗m)

Definition at line 164 of file aero_drag.hh.

Referenced by aero_drag().

8.3.4.5 ballistic_drag

```
DefaultAero jeod::AerodynamicDrag::ballistic_drag
```

Spherical, ballistic drag.

The default, default behavior. Can be overridden by resetting the "default_behavior" pointertrick_units(–)

Definition at line 193 of file aero_drag.hh.

8.3.4.6 constant_density

```
bool jeod::AerodynamicDrag::constant_density {}
```

Use constant density for aero drag?

trick_units(–)

Definition at line 147 of file aero_drag.hh.

Referenced by aero_drag().

8.3.4.7 default_behavior

`DefaultAero* jeod::AerodynamicDrag::default_behavior`

Pointer to an object that defines the default aero behavior.

This is used if the AeroSurface pointer in aero_drag is set to NULL. Defaults to ballistic drag, but can be overriddentrick_units(--)

Definition at line 186 of file aero_drag.hh.

Referenced by aero_drag().

8.3.4.8 density

`double jeod::AerodynamicDrag::density {}`

Density of the last time AerodynamicDrag was used.

If constant_density is set true, then this is the density that will be usedtrick_units(kg/m3)

Definition at line 154 of file aero_drag.hh.

Referenced by aero_drag().

8.3.4.9 param

`AeroDragParameters jeod::AerodynamicDrag::param`

parameters shared with plate model

trick_units(--)

Definition at line 169 of file aero_drag.hh.

Referenced by aero_drag(), and jeod::AerodynamicDrag_aero_model_default_data::initialize().

8.3.4.10 use_default_behavior

`bool jeod::AerodynamicDrag::use_default_behavior {true}`

Use the default behavior?

trick_units(--)

Definition at line 174 of file aero_drag.hh.

Referenced by aero_drag().

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

- aero_drag.hh
- aero_drag.cc

8.4 jeod::AerodynamicDrag_aero_model_default_data Class Reference

```
#include <aero_model.hh>
```

Public Member Functions

- void initialize (AerodynamicDrag ∗)

8.4.1 Detailed Description

Definition at line 55 of file aero_model.hh.

8.4.2 Member Function Documentation

8.4.2.1 initialize()

```
void jeod::AerodynamicDrag_aero_model_default_data::initialize (
            AerodynamicDrag * AerodynamicDrag_ptr )
```

Definition at line 28 of file aero_model.cc.

References jeod::AerodynamicDrag::active, jeod::AeroDragParameters::gas_const, jeod::AerodynamicDrag↩ ::param, and jeod::AeroDragParameters::temp_free_stream.

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

- aero_model.hh
- aero_model.cc

8.5 jeod::AerodynamicsMessages Class Reference

Messages associated with use of the aerodynamics model.

```
#include <aerodynamics_messages.hh>
```

Public Member Functions

- AerodynamicsMessages ()=delete
- AerodynamicsMessages (const AerodynamicsMessages &rhs)=delete
- AerodynamicsMessages & operator= (const AerodynamicsMessages &rhs)=delete

Static Public Attributes

- static const char ∗ initialization_error = "interactions/aerodynamics/" "initialization_error"

  Associated with errors during initialization of the drag model.
- static const char ∗ runtime_error = "interactions/aerodynamics/" "runtime_error"

  Associated with errors during the runtime of the drag model.
- static const char ∗ pre_initialization_error = "interactions/aerodynamics/" "pre_initialization_error"

  Associated with errors during the setup of the system, before runtime.
- static const char ∗ runtime_warns = "interactions/aerodynamics/" "runtime_warns"

  Associated with warnings given at runtime.

Friends

- class InputProcessor
- void init_attrjeod__AerodynamicsMessages ()

8.5.1 Detailed Description

Messages associated with use of the aerodynamics model.

Definition at line 85 of file aerodynamics_messages.hh.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 AerodynamicsMessages() [1/2]

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

8.5.2.2 AerodynamicsMessages() [2/2]

```
jeod::AerodynamicsMessages::AerodynamicsMessages (
             const AerodynamicsMessages & rhs )   [delete]
```

8.5.3 Member Function Documentation

8.5.3.1 operator=()

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

8.5.4 Friends And Related Function Documentation

8.5.4.1 init_attrjeod__AerodynamicsMessages

```
void init_attrjeod__AerodynamicsMessages ( )  [friend]
```

8.5.4.2 InputProcessor

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

Definition at line 87 of file aerodynamics_messages.hh.

8.5.5 Field Documentation

8.5.5.1 initialization_error

```
const char * jeod::AerodynamicsMessages::initialization_error = "interactions/aerodynamics/"
"initialization_error"  [static]
```

Associated with errors during initialization of the drag model.

trick_units(–)

Definition at line 95 of file aerodynamics_messages.hh.

Referenced by jeod::AeroSurface::allocate_array(), jeod::AeroSurface::allocate_interaction_facet(), and jeod::Flat↩ PlateAeroFactory::create_facet().

8.5.5.2 pre_initialization_error

```
const char * jeod::AerodynamicsMessages::pre_initialization_error = "interactions/aerodynamics/"
"pre_initialization_error"  [static]
```

Associated with errors during the setup of the system, before runtime.

trick_units(–)

Definition at line 103 of file aerodynamics_messages.hh.

Referenced by jeod::AeroSurfaceFactory::add_facet_params().

8.5.5.3 runtime_error

```
const char * jeod::AerodynamicsMessages::runtime_error = "interactions/aerodynamics/" "runtime←
_error" [static]
```

Associated with errors during the runtime of the drag model.

trick_units(–)

Definition at line 99 of file aerodynamics_messages.hh.

Referenced by jeod::AerodynamicDrag::aero_drag(), jeod::FlatPlateAeroFacet::aerodrag_force(), and jeod::←
DefaultAero::aerodrag_force().

8.5.5.4 runtime_warns

```
const char * jeod::AerodynamicsMessages::runtime_warns = "interactions/aerodynamics/" "runtime←
_warns" [static]
```

Associated with warnings given at runtime.

trick_units(–)

Definition at line 110 of file aerodynamics_messages.hh.

Referenced by jeod::FlatPlateAeroFacet::aerodrag_force(), and jeod::DefaultAero::aerodrag_force().

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

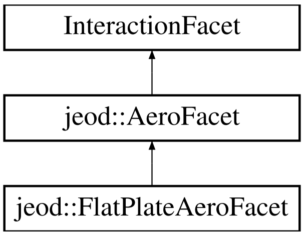
- aerodynamics_messages.hh
- aerodynamics_messages.cc

8.6 jeod::AeroFacet Class Reference

An aerodynamic interaction specific facet for use in the surface model.

```
#include <aero_facet.hh>
```

Inheritance diagram for jeod::AeroFacet:



Public Member Functions

- AeroFacet ()=default
- ∼AeroFacet () override=default
- AeroFacet & operator= (const AeroFacet &)=delete
- AeroFacet (const AeroFacet &)=delete
- virtual void aerodrag_force (const double velocity_mag, const double rel_vel_hat[3], AeroDragParameters ∗aero_drag_param_ptr, double center_grav[3])=0

  A pure virtual function defining the interface for all aerodynamic interaction facets.

Friends

- class InputProcessor
- void init_attrjeod__AeroFacet ()

8.6.1 Detailed Description

An aerodynamic interaction specific facet for use in the surface model.

Definition at line 77 of file aero_facet.hh.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 AeroFacet() [1/2]

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

8.6.2.2 ∼AeroFacet()

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

8.6.2.3 AeroFacet() [2/2]

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

8.6.3 Member Function Documentation

8.6.3.1 aerodrag_force()

```
virtual void jeod::AeroFacet::aerodrag_force (
             const double velocity_mag,
             const double rel_vel_hat[3],
             AeroDragParameters ∗ aero_drag_param_ptr,
             double center_grav[3] )  [pure virtual]
```

A pure virtual function defining the interface for all aerodynamic interaction facets.

All aerodynamic interaction facets inherited from AeroFacet must implement this function

Parameters

in	*velocity_mag*	The magnitude of the relative inertial velocity, including wind, of the vehicle Units: m/s
in	*rel_vel_hat*	The Unit vector of the relative inertial velocity
in	*aero_drag_param_ptr*	The parameters used to calculate aerodynamic drag
in	*center_grav*	The position of the center of graviy of the vehicle, in the structural frame Units: m

Implemented in jeod::FlatPlateAeroFacet.

Referenced by jeod::AerodynamicDrag::aero_drag().

8.6.3.2 operator=()

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

8.6.4 Friends And Related Function Documentation

8.6.4.1 init_attrjeod__AeroFacet

```
void init_attrjeod__AeroFacet ( )  [friend]
```

8.6.4.2 InputProcessor

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

Definition at line 79 of file aero_facet.hh.

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

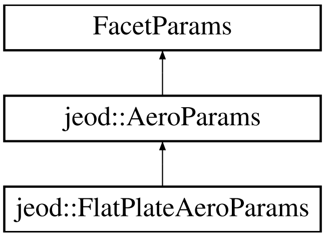
- aero_facet.hh

8.7 jeod::AeroParams Class Reference

A base class for all aerodynamic parameters used in the surface model.

`#include <aero_params.hh>`

Inheritance diagram for jeod::AeroParams:



Public Member Functions

- AeroParams ()=default
- ∼AeroParams () override=default
- AeroParams & operator= (const AeroParams &)=delete
- AeroParams (const AeroParams &)=delete

Friends

- class InputProcessor
- void init_attrjeod__AeroParams ()

8.7.1 Detailed Description

A base class for all aerodynamic parameters used in the surface model.

Definition at line 76 of file aero_params.hh.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 AeroParams() [1/2]

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

8.7.2.2 ∼AeroParams()

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

8.7.2.3 AeroParams() [2/2]

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

8.7.3 Member Function Documentation

8.7.3.1 operator=()

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

8.7.4 Friends And Related Function Documentation

8.7.4.1 init_attrjeod__AeroParams

```
void init_attrjeod__AeroParams ( )  [friend]
```

8.7.4.2 InputProcessor

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

Definition at line 78 of file aero_params.hh.

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

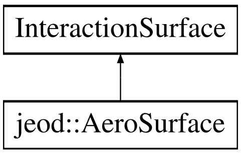
- aero_params.hh

8.8 jeod::AeroSurface Class Reference

The aerodynamic specific interaction surface, for use with the surface model.

```
#include <aero_surface.hh>
```

Inheritance diagram for jeod::AeroSurface:



Public Member Functions

- AeroSurface ()

  Default Constructor.
- ∼AeroSurface () override

  Destructor.
- AeroSurface & operator= (const AeroSurface &)=delete
- AeroSurface (const AeroSurface &)=delete
- void allocate_array (unsigned int size) override

  Allocates an array of AeroFacet pointers, of the size indicated by the input variable.
- void allocate_interaction_facet (Facet ∗facet, InteractionFacetFactory ∗factory, FacetParams ∗params, unsigned int index) override

  Allocates a particular interaction facet, from an inputted general facet, using the inputted parameters and interaction facet factory.

Data Fields

- AeroFacet ∗∗ aero_facets {}

  An array of pointers to aerodynamic interaction facets.
- unsigned int facets_size {}

  Size of the aero_facets array.

Friends

- class InputProcessor
- void init_attrjeod__AeroSurface ()

8.8.1 Detailed Description

The aerodynamic specific interaction surface, for use with the surface model.

Definition at line 85 of file aero_surface.hh.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 AeroSurface() [1/2]

```
jeod::AeroSurface::AeroSurface ( )
```

Default Constructor.

Definition at line 53 of file aero_surface.cc.

8.8.2.2 ∼AeroSurface()

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

Destructor.

Definition at line 63 of file aero_surface.cc.

References aero_facets, and facets_size.

8.8.2.3 AeroSurface() [2/2]

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

8.8.3 Member Function Documentation

8.8.3.1 allocate_array()

```
void jeod::AeroSurface::allocate_array (
             unsigned int size )  [override]
```

Allocates an array of AeroFacet pointers, of the size indicated by the input variable.

Parameters

in	*size*	The size of the needed array Units: cnt:

Definition at line 74 of file aero_surface.cc.

References aero_facets, facets_size, and jeod::AerodynamicsMessages::initialization_error.

8.8.3.2 allocate_interaction_facet()

```
void jeod::AeroSurface::allocate_interaction_facet (
            Facet * facet,
            InteractionFacetFactory * factory,
            FacetParams * params,
            unsigned int index )  [override]
```

Allocates a particular interaction facet, from an inputted general facet, using the inputted parameters and interaction facet factory.

This facet is then placed at the index given. If the correct InteractionFacetFactory and Facet Params are not given for the aerodynamic interaction or for the type of facet given, a fail message will be sent

Parameters

in	*facet*	The basic facet used to create the interaction facet
in	*factory*	The factory used to create the interaction facet
in	*params*	The aero params used to create the interaction facet
in	*index*	Where the new interaction facet will be placed in the aero_facets array Units: cnt

Definition at line 123 of file aero_surface.cc.

References aero_facets, facets_size, and jeod::AerodynamicsMessages::initialization_error.

8.8.3.3 operator=()

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

8.8.4 Friends And Related Function Documentation

8.8.4.1 init_attrjeod__AeroSurface

```
void init_attrjeod__AeroSurface ( )  [friend]
```

8.8.4.2 InputProcessor

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

Definition at line 87 of file aero_surface.hh.

8.8.5 Field Documentation

8.8.5.1 aero_facets

```
AeroFacet** jeod::AeroSurface::aero_facets {}
```

An array of pointers to aerodynamic interaction facets.

AeroFacets is a pure virtual, so these will all be pointed to inheriting classes through polymorphismtrick_units(–)

Definition at line 98 of file aero_surface.hh.

Referenced by jeod::AerodynamicDrag::aero_drag(), allocate_array(), allocate_interaction_facet(), and ∼AeroSurface().

8.8.5.2 facets_size

```
unsigned int jeod::AeroSurface::facets_size {}
```

Size of the aero_facets array.

trick_units(count)

Definition at line 103 of file aero_surface.hh.

Referenced by jeod::AerodynamicDrag::aero_drag(), allocate_array(), allocate_interaction_facet(), and ∼AeroSurface().

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

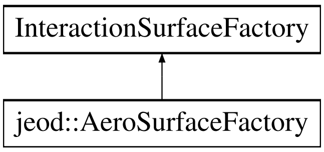
- aero_surface.hh
- aero_surface.cc

8.9 jeod::AeroSurfaceFactory Class Reference

The surface factory that creates an aerodynamic specific surface from a general surface.

```
#include <aero_surface_factory.hh>
```

Inheritance diagram for jeod::AeroSurfaceFactory:



Public Member Functions

- AeroSurfaceFactory ()

  Default Constructor.
- ∼AeroSurfaceFactory () override=default
- AeroSurfaceFactory & operator= (const AeroSurfaceFactory &)=delete
- AeroSurfaceFactory (const AeroSurfaceFactory &)=delete
- void add_facet_params (FacetParams ∗to_add) override

  Add a named set of facet params to the surface factory.

Protected Attributes

- FlatPlateAeroFactory flat_plate_aero_factory

  A factory that can create a flat plate aero facet from a flat plate.
- FlatPlateThermalAeroFactory flat_plate_thermal_aero_factory

  A factory that can create a flat plate aero facet from a flat plate.

Friends

- class InputProcessor
- void init_attrjeod__AeroSurfaceFactory ()

8.9.1 Detailed Description

The surface factory that creates an aerodynamic specific surface from a general surface.

Used with the surface model.

Definition at line 87 of file aero_surface_factory.hh.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 AeroSurfaceFactory() [1/2]

```
jeod::AeroSurfaceFactory::AeroSurfaceFactory ( )
```

Default Constructor.

Definition at line 47 of file aero_surface_factory.cc.

References flat_plate_aero_factory, and flat_plate_thermal_aero_factory.

8.9.2.2 ∼AeroSurfaceFactory()

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

8.9.2.3 AeroSurfaceFactory() [2/2]

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

8.9.3 Member Function Documentation

8.9.3.1 add_facet_params()

```
void jeod::AeroSurfaceFactory::add_facet_params (
            FacetParams * to_add )  [override]
```

Add a named set of facet params to the surface factory.

Intended to be used when an aerodynamic specific surface is created, to convert a basic facet to an aerodynamic interaction facet. This MUST be a parameter inheriting from AeroParam, or the function will fail and send a failure message

Parameters

in	*to_add*	The facet parameters to add

Definition at line 64 of file aero_surface_factory.cc.

References jeod::AerodynamicsMessages::pre_initialization_error.

8.9.3.2 operator=()

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

8.9.4 Friends And Related Function Documentation

8.9.4.1 init_attrjeod__AeroSurfaceFactory

```
void init_attrjeod__AeroSurfaceFactory ( )  [friend]
```

8.9.4.2 InputProcessor

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

Definition at line 89 of file aero_surface_factory.hh.

8.9.5 Field Documentation

8.9.5.1 flat_plate_aero_factory

```
FlatPlateAeroFactory jeod::AeroSurfaceFactory::flat_plate_aero_factory  [protected]
```

A factory that can create a flat plate aero facet from a flat plate.

trick_units(--)

Definition at line 107 of file aero_surface_factory.hh.

Referenced by AeroSurfaceFactory().

8.9.5.2 flat_plate_thermal_aero_factory

```
FlatPlateThermalAeroFactory jeod::AeroSurfaceFactory::flat_plate_thermal_aero_factory  [protected]
```

A factory that can create a flat plate aero facet from a flat plate.

trick_units(--)

Definition at line 111 of file aero_surface_factory.hh.

Referenced by AeroSurfaceFactory().

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

- aero_surface_factory.hh
- aero_surface_factory.cc

8.10 jeod::DefaultAero Class Reference

The simple, default, aerodynamic drag model, including coefficient of drag, ballistic coefficient, etc.

`#include <default_aero.hh>`

Public Types

- enum DragOption { DRAG_OPT_CD = 0, DRAG_OPT_BC = 1, DRAG_OPT_CONST = 2 }

  Specifies how drag is to be computed.

Public Member Functions

- DefaultAero ()=default
- virtual ∼DefaultAero ()=default
- DefaultAero & operator= (const DefaultAero &)=delete
- DefaultAero (const DefaultAero &)=delete
- virtual void aerodrag_force (const double velocity_mag, const double rel_vel_hat[3], AeroDragParameters ∗aero_drag_param_ptr, double mass, double force[3], double torque[3])

  The implementation for this aerodynamic drags force and torque calculations.

Data Fields

- double Cd {}

  Coefficient of drag.
- double BC {}

  Ballistic Coefficient.
- double area {}

  Vehicle aerodynamic area.
- double drag {}

  Drag calculated during use.
- DragOption option {DRAG_OPT_CONST}

  The type of simple drag to use.

Friends

- class InputProcessor
- void init_attrjeod__DefaultAero ()

8.10.1 Detailed Description

The simple, default, aerodynamic drag model, including coefficient of drag, ballistic coefficient, etc.

This can be overriden with a user defined model in the AerodynamicDrag class.

Definition at line 86 of file default_aero.hh.

8.10.2 Member Enumeration Documentation

8.10.2.1 DragOption

`enum jeod::DefaultAero::DragOption`

Specifies how drag is to be computed.

Enumerator

DRAG_OPT_CD	Use Coefficient of drag for drag computations.
DRAG_OPT_BC	Use Ballistic Coefficient for drag computations.
DRAG_OPT_CONST	Use specified constant drag.

Definition at line 92 of file default_aero.hh.

8.10.3 Constructor & Destructor Documentation

8.10.3.1 DefaultAero() [1/2]

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

8.10.3.2 ∼DefaultAero()

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

8.10.3.3 DefaultAero() [2/2]

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

8.10.4 Member Function Documentation

8.10.4.1 aerodrag_force()

```
void jeod::DefaultAero::aerodrag_force (
            const double velocity_mag,
            const double rel_vel_hat[3],
            AeroDragParameters * aero_drag_param_ptr,
            double mass,
            double force[3],
            double torque[3] )  [virtual]
```

The implementation for this aerodynamic drags force and torque calculations.

Can be overriden by an inheriting class to create extensibility

Parameters

in	*velocity_mag*	The magnitude of the relative velocity of the vehicle; not used here but some child classes need it Units: M/s
in	*rel_vel_hat*	The unit vector of the relative velocity of the vehicle, in the structural frame
in	*aero_drag_param_ptr*	The aerodynamic drag parameters used to calculate drag
in	*mass*	The current mass of the vehicle Units: kg
out	*force*	The aerodynamic force, in the structural frame Units: N
out	*torque*	The aerodynamic torque, in the structural frame Units: N∗M

Definition at line 59 of file default_aero.cc.

References area, BC, Cd, drag, DRAG_OPT_BC, DRAG_OPT_CD, DRAG_OPT_CONST, jeod::AeroDragParameters::dynamic_pressure, option, jeod::AerodynamicsMessages::runtime_error, and jeod::AerodynamicsMessages::runtime_warns.

Referenced by jeod::AerodynamicDrag::aero_drag().

8.10.4.2 operator=()

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

8.10.5 Friends And Related Function Documentation

8.10.5.1 init_attrjeod__DefaultAero

```
void init_attrjeod__DefaultAero ( )  [friend]
```

8.10.5.2 InputProcessor

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

Definition at line 88 of file default_aero.hh.

8.10.6 Field Documentation

8.10.6.1 area

```
double jeod::DefaultAero::area {}
```

Vehicle aerodynamic area.

trick_units(m2)

Definition at line 124 of file default_aero.hh.

Referenced by aerodrag_force().

8.10.6.2 BC

```
double jeod::DefaultAero::BC {}
```

Ballistic Coefficient.

trick_units(kg/m2)

Definition at line 120 of file default_aero.hh.

Referenced by aerodrag_force().

8.10.6.3 Cd

```
double jeod::DefaultAero::Cd {}
```

Coefficient of drag.

trick_units(–)

Definition at line 116 of file default_aero.hh.

Referenced by aerodrag_force().

8.10.6.4 drag

```
double jeod::DefaultAero::drag {}
```

Drag calculated during use.

Can be set by user and will then never be changed with a DRAG_OPT_CONSTtrick_units(N)

Definition at line 130 of file default_aero.hh.

Referenced by aerodrag_force().

8.10.6.5 option

```
DragOption jeod::DefaultAero::option {DRAG_OPT_CONST}
```

The type of simple drag to use.

trick_units(–)

Definition at line 135 of file default_aero.hh.

Referenced by aerodrag_force().

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

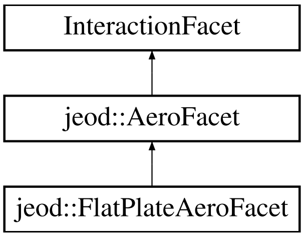
- default_aero.hh
- default_aero.cc

8.11 jeod::FlatPlateAeroFacet Class Reference

The aerdynamic specific version of a flat plate.

```
#include <flat_plate_aero_facet.hh>
```

Inheritance diagram for jeod::FlatPlateAeroFacet:



Public Member Functions

- FlatPlateAeroFacet ()=default
- ∼FlatPlateAeroFacet () override=default
- FlatPlateAeroFacet & operator= (const FlatPlateAeroFacet &)=delete
- FlatPlateAeroFacet (const FlatPlateAeroFacet &)=delete
- void aerodrag_force (const double velocity_mag, const double rel_vel_hat[3], AeroDragParameters ∗aero_drag_param_ptr, double center_grav[3]) override

  The FlatPlateAeroFacet specific implementation of aerodynamic drag force, based on the given parameters.

Data Fields

- double ∗ center_pressure {}

  Flat plate center of pressure (in structural frame).
- double ∗ normal {}

  Unit vector normal to the plate surface, pointing outward (structural frame).
- double force_n {}

  Magnitude of the force normal to the plate.
- double force_t {}

  Magnitude of the force tangential to the plate, or parallel to the velocity vector, depending on application.
- AeroDragEnum::CoefCalcMethod coef_method {AeroDragEnum::Specular}

  Enum indicating which method of coef calculation to use: specular, diffuse, calculated, mixed.
- bool calculate_drag_coef {}

  whether to calculate the drag coefficient
- double epsilon {}

  fraction of molecules that "bounce"
- double temp_reflect {}

  temperature of reflected molecules
- double drag_coef_norm {}

  The coefficient for calculating drag normal to the plate.
- double drag_coef_tang {}

  The coefficient for calculating drag tangential to the plate.
- double drag_coef_spec {}

  The coefficient for calculating drag resulting only from molecules bouncing off the surface.
- double drag_coef_diff {}

  The coefficient for calculating drag resulting only from molecules sticking to the surface.
- double temperature {}

  Temperature of the plate.
- double area {}

  area of the plate

Friends

- class InputProcessor
- void init_attrjeod__FlatPlateAeroFacet ()

8.11.1 Detailed Description

The aerdynamic specific version of a flat plate.

Definition at line 84 of file flat_plate_aero_facet.hh.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 FlatPlateAeroFacet() [1/2]

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

8.11.2.2 ∼FlatPlateAeroFacet()

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

8.11.2.3 FlatPlateAeroFacet() [2/2]

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

8.11.3 Member Function Documentation

8.11.3.1 aerodrag_force()

```
void jeod::FlatPlateAeroFacet::aerodrag_force (
            const double rel_vel_mag,
            const double rel_vel_struct_hat[3],
            AeroDragParameters * aero_drag_param_ptr,
            double center_grav[3] )  [override], [virtual]
```

The FlatPlateAeroFacet specific implementation of aerodynamic drag force, based on the given parameters.

Parameters

in	*rel_vel_mag*	The magnitude of the relative velocity Units: M/s
in	*rel_vel_struct_hat*	The unit vector of the total relative velocity, in the structural frame
in	*aero_drag_param_ptr*	The aerodynamic drag parameters used for drag calculation
in	*center_grav*	The center of gravity of the vehicle, in the structural frame Units: M

Implements jeod::AeroFacet.

Definition at line 49 of file flat_plate_aero_facet.cc.

References area, jeod::AeroDragEnum::Calc_coef, calculate_drag_coef, center_pressure, coef_method, jeod::AeroDragEnum::Diffuse, drag_coef_diff, drag_coef_norm, drag_coef_spec, drag_coef_tang, jeod::AeroDragParameters::dynamic_pressure, epsilon, force_n, force_t, jeod::AeroDragParameters::gas_const, jeod::AeroDragEnum::Mixed, normal, jeod::AerodynamicsMessages::runtime_error, jeod::AerodynamicsMessages::runtime_warns, jeod::AeroDragEnum::Specular, jeod::AeroDragParameters::temp_free_stream, and temperature.

8.11.3.2 operator=()

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

8.11.4 Friends And Related Function Documentation

8.11.4.1 init_attrjeod__FlatPlateAeroFacet

```
void init_attrjeod__FlatPlateAeroFacet ( )  [friend]
```

8.11.4.2 InputProcessor

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

Definition at line 86 of file flat_plate_aero_facet.hh.

8.11.5 Field Documentation

8.11.5.1 area

```
double jeod::FlatPlateAeroFacet::area {}
```

area of the plate

trick_units(m2)

Definition at line 174 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.2 calculate_drag_coef

```
bool jeod::FlatPlateAeroFacet::calculate_drag_coef {}
```

whether to calculate the drag coefficient

trick_units(–)

Definition at line 132 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.3 center_pressure

```
double* jeod::FlatPlateAeroFacet::center_pressure {}
```

Flat plate center of pressure (in structural frame).

Once the aero surface is initialized, it points to the position found in FlatPlatetrick_units(m)

Definition at line 103 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.4 coef_method

```
AeroDragEnum::CoefCalcMethod jeod::FlatPlateAeroFacet::coef_method {AeroDragEnum::Specular}
```

Enum indicating which method of coef calculation to use: specular, diffuse, calculated, mixed.

trick_units(–)

Definition at line 127 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.5 drag_coef_diff

```
double jeod::FlatPlateAeroFacet::drag_coef_diff {}
```

The coefficient for calculating drag resulting only from molecules sticking to the surface.

trick_units(–)

Definition at line 164 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.6 drag_coef_norm

```
double jeod::FlatPlateAeroFacet::drag_coef_norm {}
```

The coefficient for calculating drag normal to the plate.

trick_units(–)

Definition at line 147 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.7 drag_coef_spec

```
double jeod::FlatPlateAeroFacet::drag_coef_spec {}
```

The coefficient for calculating drag resulting only from molecules bouncing off the surface.

trick_units(–)

Definition at line 158 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.8 drag_coef_tang

```
double jeod::FlatPlateAeroFacet::drag_coef_tang {}
```

The coefficient for calculating drag tangential to the plate.

trick_units(–)

Definition at line 152 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.9 epsilon

```
double jeod::FlatPlateAeroFacet::epsilon {}
```

fraction of molecules that "bounce"

trick_units(–)

Definition at line 137 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.10 force_n

```
double jeod::FlatPlateAeroFacet::force_n {}
```

Magnitude of the force normal to the plate.

trick_units(N)

Definition at line 115 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force().

8.11.5.11 force_t

```
double jeod::FlatPlateAeroFacet::force_t {}
```

Magnitude of the force tangential to the plate, or parallel to the velocity vector, depending on application.

trick_units(N)

Definition at line 121 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force().

8.11.5.12 normal

```
double* jeod::FlatPlateAeroFacet::normal {}
```

Unit vector normal to the plate surface, pointing outward (structural frame).

Once the aero surface is initialized, it points to the normal found in FlatPlatetrick_units(–)

Definition at line 110 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

8.11.5.13 temp_reflect

```
double jeod::FlatPlateAeroFacet::temp_reflect {}
```

temperature of reflected molecules

trick_units(K)

Definition at line 142 of file flat_plate_aero_facet.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.11.5.14 temperature

```
double jeod::FlatPlateAeroFacet::temperature {}
```

Temperature of the plate.

trick_units(K)

Definition at line 169 of file flat_plate_aero_facet.hh.

Referenced by aerodrag_force(), and jeod::FlatPlateAeroFactory::create_facet().

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

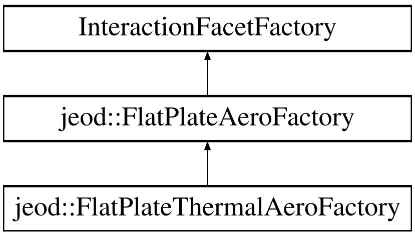
- flat_plate_aero_facet.hh
- flat_plate_aero_facet.cc

8.12 jeod::FlatPlateAeroFactory Class Reference

Creates a FlatPlateAeroFacet from a FlatPlate.

```
#include <flat_plate_aero_factory.hh>
```

Inheritance diagram for jeod::FlatPlateAeroFactory:



Public Member Functions

- FlatPlateAeroFactory ()

  Default Constructor.
- ∼FlatPlateAeroFactory () override=default
- FlatPlateAeroFactory & operator= (const FlatPlateAeroFactory &)=delete
- FlatPlateAeroFactory (const FlatPlateAeroFactory &)=delete
- InteractionFacet ∗ create_facet (Facet ∗facet, FacetParams ∗params) override

  Create a FlatPlateAeroFacet from a flat plate facet and a FlatePlateAeroParams object.
- bool is_correct_factory (Facet ∗facet) override

  FlatPlateAeroFactory specific implementation of this function.

Friends

- class InputProcessor
- void init_attrjeod__FlatPlateAeroFactory ()

8.12.1 Detailed Description

Creates a FlatPlateAeroFacet from a FlatPlate.

Definition at line 83 of file flat_plate_aero_factory.hh.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 FlatPlateAeroFactory() [1/2]

```
jeod::FlatPlateAeroFactory::FlatPlateAeroFactory ( )
```

Default Constructor.

Definition at line 53 of file flat_plate_aero_factory.cc.

8.12.2.2 ∼FlatPlateAeroFactory()

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

8.12.2.3 FlatPlateAeroFactory() [2/2]

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

8.12.3 Member Function Documentation

8.12.3.1 create_facet()

```
InteractionFacet * jeod::FlatPlateAeroFactory::create_facet (
             Facet * facet,
             FacetParams * params )  [override]
```

Create a FlatPlateAeroFacet from a flat plate facet and a FlatePlateAeroParams object.

Returns

The new FlatPlateAeroFacet. Note that this is allocated and YOU are responsible for destroying it at the end!

Parameters

in	*facet*	The FlatPlate. This MUST be a flat plate or the algorithm will send a failure message
in	*params*	FlatPlateAeroParams. This MUST be of the type FlatPlateAeroParams, or the algorithm will send a failture message Units: The

Definition at line 68 of file flat_plate_aero_factory.cc.

References jeod::FlatPlateAeroFacet::area, jeod::FlatPlateAeroParams::calculate_drag_coef, jeod::FlatPlateAeroFacet::calculate_drag_coef, jeod::FlatPlateAeroFacet::center_pressure, jeod::FlatPlateAeroParams::coef_method, jeod::FlatPlateAeroFacet::coef_method, jeod::FlatPlateAeroParams::drag_coef_diff, jeod::FlatPlateAeroFacet::drag_coef_diff, jeod::FlatPlateAeroParams::drag_coef_norm, jeod::FlatPlateAeroFacet::drag_coef_norm, jeod::FlatPlateAeroParams::drag_coef_spec, jeod::FlatPlateAeroFacet::drag_coef_spec, jeod::FlatPlateAeroParams::drag_coef_tang, jeod::FlatPlateAeroFacet::drag_coef_tang, jeod::FlatPlateAeroParams::epsilon, jeod::FlatPlateAeroFacet::epsilon, jeod::AerodynamicsMessages::initialization_error, jeod::FlatPlateAeroFacet::normal, jeod::FlatPlateAeroParams::temp_reflect, jeod::FlatPlateAeroFacet::temp_reflect, and jeod::FlatPlateAeroFacet::temperature.

8.12.3.2 is_correct_factory()

```
bool jeod::FlatPlateAeroFactory::is_correct_factory (
             Facet * facet )  [override]
```

FlatPlateAeroFactory specific implementation of this function.

If the Facet is of type FlatPlate, returns true. False otherwise

Returns

true if facet is a FlatPlate, false otherwise

Parameters

in	*facet*	The facet to check

Definition at line 134 of file flat_plate_aero_factory.cc.

8.12.3.3 operator=()

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

8.12.4 Friends And Related Function Documentation

8.12.4.1 init_attrjeod__FlatPlateAeroFactory

```
void init_attrjeod__FlatPlateAeroFactory ( )  [friend]
```

8.12.4.2 InputProcessor

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

Definition at line 85 of file flat_plate_aero_factory.hh.

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

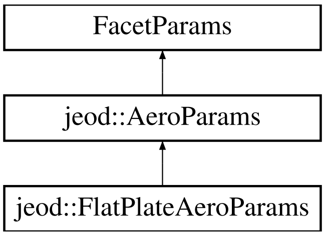
- flat_plate_aero_factory.hh
- flat_plate_aero_factory.cc

8.13 jeod::FlatPlateAeroParams Class Reference

used in the FlatPlateAeroFactory to create a FlatPlateAeroFacet.

```
#include <flat_plate_aero_params.hh>
```

Inheritance diagram for jeod::FlatPlateAeroParams:



Public Member Functions

- FlatPlateAeroParams ()=default
- ∼FlatPlateAeroParams () override=default
- FlatPlateAeroParams & operator= (const FlatPlateAeroParams &)=delete
- FlatPlateAeroParams (const FlatPlateAeroParams &)=delete

Data Fields

- AeroDragEnum::CoefCalcMethod coef_method {AeroDragEnum::Specular}

  Enum indicating which method of coef calculation to use: specular, diffuse, calculated, mixed.
- bool calculate_drag_coef {}

  Whether to calculate the drag coefficient.
- double epsilon {}

  Fraction of molecules that "bounce".
- double temp_reflect {}

  Temperature of reflected molecules.
- double drag_coef_norm {}

  The coefficient for calculating drag normal to the plate.
- double drag_coef_tang {}

  The coefficient for calculating drag tangential to the plate.
- double drag_coef_spec {}

  The coefficient for calculating drag resulting only from molecules bouncing off the surface.
- double drag_coef_diff {}

  The coefficient for calculating drag resulting only from molecules sticking to the surface.

Friends

- class InputProcessor
- void init_attrjeod__FlatPlateAeroParams ()

8.13.1 Detailed Description

used in the FlatPlateAeroFactory to create a FlatPlateAeroFacet.

Definition at line 76 of file flat_plate_aero_params.hh.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 FlatPlateAeroParams() [1/2]

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

8.13.2.2 ∼FlatPlateAeroParams()

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

8.13.2.3 FlatPlateAeroParams() [2/2]

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

8.13.3 Member Function Documentation

8.13.3.1 operator=()

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

8.13.4 Friends And Related Function Documentation

8.13.4.1 init_attrjeod__FlatPlateAeroParams

```
void init_attrjeod__FlatPlateAeroParams ( )  [friend]
```

8.13.4.2 InputProcessor

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

Definition at line 78 of file flat_plate_aero_params.hh.

8.13.5 Field Documentation

8.13.5.1 calculate_drag_coef

```
bool jeod::FlatPlateAeroParams::calculate_drag_coef {}
```

Whether to calculate the drag coefficient.

trick_units(–)

Definition at line 93 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.2 coef_method

```
AeroDragEnum::CoefCalcMethod jeod::FlatPlateAeroParams::coef_method {AeroDragEnum::Specular}
```

Enum indicating which method of coef calculation to use: specular, diffuse, calculated, mixed.

trick_units(–)

Definition at line 88 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.3 drag_coef_diff

```
double jeod::FlatPlateAeroParams::drag_coef_diff {}
```

The coefficient for calculating drag resulting only from molecules sticking to the surface.

trick_units(–)

Definition at line 125 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.4 drag_coef_norm

```
double jeod::FlatPlateAeroParams::drag_coef_norm {}
```

The coefficient for calculating drag normal to the plate.

trick_units(–)

Definition at line 108 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.5 drag_coef_spec

```
double jeod::FlatPlateAeroParams::drag_coef_spec {}
```

The coefficient for calculating drag resulting only from molecules bouncing off the surface.

trick_units(–)

Definition at line 119 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.6 drag_coef_tang

```
double jeod::FlatPlateAeroParams::drag_coef_tang {}
```

The coefficient for calculating drag tangential to the plate.

trick_units(–)

Definition at line 113 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.7 epsilon

```
double jeod::FlatPlateAeroParams::epsilon {}
```

Fraction of molecules that "bounce".

trick_units(–)

Definition at line 98 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

8.13.5.8 temp_reflect

```
double jeod::FlatPlateAeroParams::temp_reflect {}
```

Temperature of reflected molecules.

trick_units(K)

Definition at line 103 of file flat_plate_aero_params.hh.

Referenced by jeod::FlatPlateAeroFactory::create_facet().

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

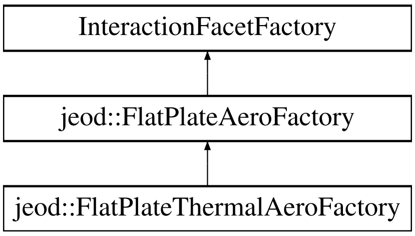
- flat_plate_aero_params.hh

8.14 jeod::FlatPlateThermalAeroFactory Class Reference

Creates a FlatPlateAeroFacet from a FlatPlate.

```
#include <flat_plate_thermal_aero_factory.hh>
```

Inheritance diagram for jeod::FlatPlateThermalAeroFactory:



Public Member Functions

- FlatPlateThermalAeroFactory ()

  Default Constructor.
- ∼FlatPlateThermalAeroFactory () override=default
- FlatPlateThermalAeroFactory & operator= (const FlatPlateThermalAeroFactory &)=delete
- FlatPlateThermalAeroFactory (const FlatPlateThermalAeroFactory &)=delete
- bool is_correct_factory (Facet ∗facet) override

  FlatPlateThermalAeroFactory specific implementation of this function.

Friends

- class InputProcessor
- void init_attrjeod__FlatPlateThermalAeroFactory ()

8.14.1 Detailed Description

Creates a FlatPlateAeroFacet from a FlatPlate.

Definition at line 81 of file flat_plate_thermal_aero_factory.hh.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 FlatPlateThermalAeroFactory() [1/2]

```
jeod::FlatPlateThermalAeroFactory::FlatPlateThermalAeroFactory ( )
```

Default Constructor.

Definition at line 48 of file flat_plate_thermal_aero_factory.cc.

8.14.2.2 ∼FlatPlateThermalAeroFactory()

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

8.14.2.3 FlatPlateThermalAeroFactory() [2/2]

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

8.14.3 Member Function Documentation

8.14.3.1 is_correct_factory()

```
bool jeod::FlatPlateThermalAeroFactory::is_correct_factory (
             Facet * facet )  [override]
```

FlatPlateThermalAeroFactory specific implementation of this function.

If the Facet is of type FlatPlate, returns true. False otherwise

Returns

true if facet is a FlatPlate, false otherwise

Parameters

in	*facet*	The facet to check

Definition at line 61 of file flat_plate_thermal_aero_factory.cc.

8.14.3.2 operator=()

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

8.14.4 Friends And Related Function Documentation

8.14.4.1 init_attrjeod__FlatPlateThermalAeroFactory

```
void init_attrjeod__FlatPlateThermalAeroFactory ( )  [friend]
```

8.14.4.2 InputProcessor

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

Definition at line 83 of file flat_plate_thermal_aero_factory.hh.

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

- flat_plate_thermal_aero_factory.hh
- flat_plate_thermal_aero_factory.cc

Chapter 9

File Documentation

9.1 aero_drag.cc File Reference

Orbital aerodynamic force and torque computation, and related classes.

```
#include <cmath>
#include <cstddef>
#include "environment/atmosphere/base_atmos/include/atmosphere.hh"
#include "environment/atmosphere/base_atmos/include/atmosphere_state.hh"
#include "utils/math/include/vector3.hh"
#include "../include/aero_drag.hh"
#include "../include/aero_facet.hh"
#include "../include/aero_surface.hh"
#include "../include/aerodynamics_messages.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.1.1 Detailed Description

Orbital aerodynamic force and torque computation, and related classes.

9.2 aero_drag.hh File Reference

Orbital aerodynamics paramter definitions and the main class for calculating aerodynamic drag.

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

Data Structures

- class jeod::AeroDragEnum

  Contains enumerations associated with aerodynamic drag.
- class jeod::AeroDragParameters

  Contains parameters associated with aerodynamic drag.
- class jeod::AerodynamicDrag

  The main class for calculating aerodynamic drag.

Namespaces

- jeod

  Namespace jeod.

9.2.1 Detailed Description

Orbital aerodynamics paramter definitions and the main class for calculating aerodynamic drag.

9.3 aero_facet.hh File Reference

Individual facets for use with aero environment interaction models.

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

Data Structures

- class jeod::AeroFacet

  An aerodynamic interaction specific facet for use in the surface model.

Namespaces

- jeod

  Namespace jeod.

9.3.1 Detailed Description

Individual facets for use with aero environment interaction models.

9.4 aero_model.cc File Reference

```
#include "interactions/aerodynamics/include/aero_drag.hh"
#include "../include/aero_model.hh"
```

Namespaces

- jeod

  Namespace jeod.

Macros

- #define JEOD_FRIEND_CLASS AerodynamicDrag_aero_model_default_data

9.4.1 Macro Definition Documentation

9.4.1.1 JEOD_FRIEND_CLASS

```
#define JEOD_FRIEND_CLASS AerodynamicDrag_aero_model_default_data
```

Definition at line 16 of file aero_model.cc.

9.5 aero_model.hh File Reference

Data Structures

- class jeod::AerodynamicDrag_aero_model_default_data

Namespaces

- jeod

  Namespace jeod.

9.6 aero_params.hh File Reference

A virtual base class for aero facet parameters, used to create interaction facets for aero in the InteractionSurface←Factorys.

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

Data Structures

- class jeod::AeroParams

  A base class for all aerodynamic parameters used in the surface model.

Namespaces

- jeod

    Namespace jeod.

9.6.1 Detailed Description

A virtual base class for aero facet parameters, used to create interaction facets for aero in the InteractionSurface↩ Factorys.

9.7 aero_surface.cc File Reference

Vehicle surface model for the aerodyanmic interaction models.

```
#include <cstddef>
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "../include/aero_facet.hh"
#include "../include/aero_surface.hh"
#include "../include/aerodynamics_messages.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.7.1 Detailed Description

Vehicle surface model for the aerodyanmic interaction models.

9.8 aero_surface.hh File Reference

Vehicle surface model for aerodynamics.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface.hh"
#include "aero_facet.hh"
#include "utils/surface_model/include/facet.hh"
```

Data Structures

- class jeod::AeroSurface

    The aerodynamic specific interaction surface, for use with the surface model.

Namespaces

- jeod

    Namespace jeod.

9.8.1 Detailed Description

Vehicle surface model for aerodynamics.

9.9 aero_surface_factory.cc File Reference

Factory that creates an aerodynamics surface, from a surface model.

```
#include <cstddef>
#include "utils/surface_model/include/facet_params.hh"
#include "../include/aero_params.hh"
#include "../include/aero_surface_factory.hh"
#include "../include/aerodynamics_messages.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.9.1 Detailed Description

Factory that creates an aerodynamics surface, from a surface model.

9.10 aero_surface_factory.hh File Reference

Factory that creates an aerodynamic interaction surface from a surface model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface_factory.hh"
#include "flat_plate_aero_factory.hh"
#include "flat_plate_thermal_aero_factory.hh"
```

Data Structures

- class jeod::AeroSurfaceFactory

    The surface factory that creates an aerodynamic specific surface from a general surface.

Namespaces

- jeod

    Namespace jeod.

9.10.1 Detailed Description

Factory that creates an aerodynamic interaction surface from a surface model.

9.11 aerodynamics_messages.cc File Reference

Implement aerodynamics_messages.

```
#include "../include/aerodynamics_messages.hh"
```

Namespaces

- jeod

    Namespace jeod.

Macros

- #define PATH "interactions/aerodynamics/"

9.11.1 Detailed Description

Implement aerodynamics_messages.

9.12 aerodynamics_messages.hh File Reference

Aerodynamics message for message handling.

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

Data Structures

- class jeod::AerodynamicsMessages

    Messages associated with use of the aerodynamics model.

Namespaces

- jeod

    Namespace jeod.

9.12.1 Detailed Description

Aerodynamics message for message handling.

9.13 class_declarations.hh File Reference

Forward declaration of classes defined in the aerodynamics package.

Namespaces

- jeod

    Namespace jeod.

9.13.1 Detailed Description

Forward declaration of classes defined in the aerodynamics package.

9.14 default_aero.cc File Reference

Implement a virtual base class for aerodynamic drag calculations.

```
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "../include/aero_drag.hh"
#include "../include/aerodynamics_messages.hh"
#include "../include/default_aero.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.14.1 Detailed Description

Implement a virtual base class for aerodynamic drag calculations.

Also implement a specific version, in the base class, that contains ballistic coefficent and coefficient of drag options

9.15 default_aero.hh File Reference

An implementation of ballistic coefficent and coefficient of drag for use in the AerodynamicDrag object.

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

Data Structures

- class jeod::DefaultAero

  The simple, default, aerodynamic drag model, including coefficient of drag, ballistic coefficient, etc.

Namespaces

- jeod

  Namespace jeod.

9.15.1 Detailed Description

An implementation of ballistic coefficent and coefficient of drag for use in the AerodynamicDrag object.

This class can be inherited from and overriden for use with the AerodynamicDrag object.

9.16 flat_plate_aero_facet.cc File Reference

Individual facets for use with aero environment interaction models.

```
#include <cstddef>
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/surface_model/include/facet.hh"
#include "../include/aerodynamics_messages.hh"
#include "../include/flat_plate_aero_facet.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.16.1 Detailed Description

Individual facets for use with aero environment interaction models.

9.17 flat_plate_aero_facet.hh File Reference

The aerodynamic specific implementation of flat plate.

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

Data Structures

- class jeod::FlatPlateAeroFacet

  The aerdynamic specific version of a flat plate.

Namespaces

- jeod

  Namespace jeod.

9.17.1 Detailed Description

The aerodynamic specific implementation of flat plate.

9.18 flat_plate_aero_factory.cc File Reference

Factory that creates a FlatPlateAeroFacet from a FlatPlate, using a FlatPlateAeroParams object.

```
#include <cstddef>
#include <typeinfo>
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/flat_plate.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "../include/aerodynamics_messages.hh"
#include "../include/flat_plate_aero_facet.hh"
#include "../include/flat_plate_aero_factory.hh"
#include "../include/flat_plate_aero_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.18.1 Detailed Description

Factory that creates a FlatPlateAeroFacet from a FlatPlate, using a FlatPlateAeroParams object.

9.19 flat_plate_aero_factory.hh File Reference

Creates a flat plate aero facet from a basic flat plate facet.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "flat_plate_aero_facet.hh"
#include "utils/surface_model/include/facet.hh"
```

Data Structures

- class jeod::FlatPlateAeroFactory

  Creates a FlatPlateAeroFacet from a FlatPlate.

Namespaces

- jeod

  Namespace jeod.

9.19.1 Detailed Description

Creates a flat plate aero facet from a basic flat plate facet.

9.20 flat_plate_aero_params.hh File Reference

The set of parameters used to create an aerodynamic interaction specific flat plate facet from a general flat plate facet.

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

Data Structures

- class jeod::FlatPlateAeroParams

  used in the FlatPlateAeroFactory to create a FlatPlateAeroFacet.

Namespaces

- jeod

    Namespace jeod.

9.20.1 Detailed Description

The set of parameters used to create an aerodynamic interaction specific flat plate facet from a general flat plate facet.

9.21 flat_plate_thermal_aero_factory.cc File Reference

Factory that creates a FlatPlateAeroFacet from a FlatPlateThermal, using a FlatPlateAeroParams object.

```
#include <typeinfo>
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/flat_plate_thermal.hh"
#include "../include/aerodynamics_messages.hh"
#include "../include/flat_plate_aero_facet.hh"
#include "../include/flat_plate_aero_params.hh"
#include "../include/flat_plate_thermal_aero_factory.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.21.1 Detailed Description

Factory that creates a FlatPlateAeroFacet from a FlatPlateThermal, using a FlatPlateAeroParams object.

9.22 flat_plate_thermal_aero_factory.hh File Reference

Creates a flat plate aero facet from a flat plate thermal facet.

```
#include "flat_plate_aero_factory.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
```

Data Structures

- class jeod::FlatPlateThermalAeroFactory

    Creates a FlatPlateAeroFacet from a FlatPlate.

Namespaces

- jeod

  Namespace jeod.

9.22.1 Detailed Description

Creates a flat plate aero facet from a flat plate thermal facet.

Index