ModelTemplateModel

5.1

Generated by Doxygen 1.8.5

Mon Jul 31 2023 11:44:56

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

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Module Documentation

5.1 Models

Modules

- Utils

5.1.1 Detailed Description

5.2 Utils

Modules

- ModelTemplate

5.2.1 Detailed Description

5.3 ModelTemplate

Files

- file foo.hh

  The file header purpose should describe why the file exists, briefly summarizing the contents of the file.
- file template_messages.hh

  Define the class TemplateMessages, the class that specifies the message IDs used in the model template model.
- file foo.cc

  Implement class Foo.
- file template_messages.cc

  Implement the class TemplateMessages.

Namespaces

- jeod

  Namespace jeod.

5.3.1 Detailed Description

Chapter 6

Namespace Documentation

6.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class Foo

  All JEOD classes should have a doxygen-style header that describes the class.
- class TemplateMessages

  Declares messages associated with the model template model.

6.1.1 Detailed Description

Namespace jeod.

Chapter 7

Data Structure Documentation

7.1 jeod::Foo Class Reference

All JEOD classes should have a doxygen-style header that describes the class.

`#include <foo.hh>`

Public Member Functions

- Foo ()

  Foo default constructor.
- ~Foo ()

  Foo destructor.
- void set_num (int value)

  Setter for Foo::num.
- int get_num ()

  Getter for Foo::num.

Protected Attributes

- int num

  A non-zero number between -42 and 42 (inclusive).

7.1.1 Detailed Description

All JEOD classes should have a doxygen-style header that describes the class.

Explain why the class exists, what it does. In this case, this class exists to demonstrate some base JEOD documentation concepts.

Definition at line 79 of file foo.hh.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 jeod::Foo::Foo (void)

Foo default constructor.

Definition at line 40 of file foo.cc.

7.1.2.2 jeod::Foo::~Foo (void)

Foo destructor.

Definition at line 52 of file foo.cc.

7.1.3 Member Function Documentation

7.1.3.1 int jeod::Foo::get_num (void)

Getter for Foo::num.

Definition at line 93 of file foo.cc.

References num.

7.1.3.2 void jeod::Foo::set_num (int *value*)

Setter for Foo::num.

Definition at line 63 of file foo.cc.

References jeod::TemplateMessages::big_negnum, num, and jeod::TemplateMessages::zero.

7.1.4 Field Documentation

7.1.4.1 int jeod::Foo::num [protected]

A non-zero number between -42 and 42 (inclusive).

trick_units(–)

Definition at line 94 of file foo.hh.

Referenced by get_num(), and set_num().

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

- foo.hh
- foo.cc

7.2 jeod::TemplateMessages Class Reference

Declares messages associated with the model template model.

`#include <template_messages.hh>`

Static Public Attributes

- static char const * big_negnum = "utils/model_template/" "big_negnum"

  Number is less than -42.
- static char const * zero = "utils/model_template/" "zero"

  Number is zero.
- static char const * big_posnum = "utils/model_template/" "big_posnum"

  Number is greater than 42.

Private Member Functions

- TemplateMessages (void)

  Not implemented.
- TemplateMessages (const TemplateMessages &)

  Not implemented.
- TemplateMessages & operator= (const TemplateMessages &)

  Not implemented.

Friends

- class InputProcessor
- void init_attrjeod__TemplateMessages ()

7.2.1 Detailed Description

Declares messages associated with the model template model.

Definition at line 83 of file template_messages.hh.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 jeod::TemplateMessages::TemplateMessages (void) [private]

Not implemented.

7.2.2.2 jeod::TemplateMessages::TemplateMessages (const TemplateMessages &) [private]

Not implemented.

7.2.3 Member Function Documentation

7.2.3.1 TemplateMessages& jeod::TemplateMessages::operator= (const TemplateMessages &) [private]

Not implemented.

7.2.4 Friends And Related Function Documentation

7.2.4.1 void init_attrjeod__TemplateMessages ( ) [friend]

7.2.4.2 friend class InputProcessor [friend]

Definition at line 86 of file template_messages.hh.

7.2.5 Field Documentation

7.2.5.1 char const * jeod::TemplateMessages::big_negnum = "utils/model_template/" "big_negnum" [static]

Number is less than -42.

trick_units(–)

Definition at line 95 of file template_messages.hh.

Referenced by jeod::Foo::set_num().

7.2.5.2 char const * jeod::TemplateMessages::big_posnum = "utils/model_template/" "big_posnum" [static]

Number is greater than 42.

trick_units(–)

Definition at line 105 of file template_messages.hh.

7.2.5.3 char const * jeod::TemplateMessages::zero = "utils/model_template/" "zero" [static]

Number is zero.

trick_units(–)

Definition at line 100 of file template_messages.hh.

Referenced by jeod::Foo::set_num().

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

- template_messages.hh
- template_messages.cc

Chapter 8

File Documentation

8.1 foo.cc File Reference

Implement class Foo.

```
#include <iostream>
#include "utils/message/include/message_handler.hh"
#include "../include/foo.hh"
#include "../include/template_messages.hh"
```

Namespaces

- jeod

  Namespace jeod.

8.1.1 Detailed Description

Implement class Foo.

Definition in file foo.cc.

8.2 foo.hh File Reference

The file header purpose should describe why the file exists, briefly summarizing the contents of the file.

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

Data Structures

- class jeod::Foo

  All JEOD classes should have a doxygen-style header that describes the class.

Namespaces

- jeod

  Namespace jeod.

8.2.1 Detailed Description

The file header purpose should describe why the file exists, briefly summarizing the contents of the file. In this case, the purpose is to define the class Foo.

Definition in file foo.hh.

8.3 template_messages.cc File Reference

Implement the class TemplateMessages.

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

Namespaces

- jeod

  Namespace jeod.

Macros

- #define MAKE_TEMPLATE_MESSAGE_CODE(id) JEOD_MAKE_MESSAGE_CODE(TemplateMessages, "utils/model_template/", id)

  The goal in a model message code implementation file is to create storage for and initialize each of static data members declared in the model message class.

8.3.1 Detailed Description

Implement the class TemplateMessages. And in this case, demonstate how to implement message code classes.

The MessageHandler defines several methods for informing the user of special conditions that can arise, ranging in severity from fatal errors to detailed debug information. Each of these methods takes a message_code argument. The intent of the message code is to identify the model that invoked the message handler and to characterize the nature of the message.

JEOD models implement that intent by making each of the message codes be a static data member of a model-specific message class. Each such data member points to a literal character string of the form "model/path/message_id", where "model/path" is the path to the model, and "message_id" is the stringified data member name.

Definition in file template_messages.cc.

8.3.2 Macro Definition Documentation

8.3.2.1 #define MAKE_TEMPLATE_MESSAGE_CODE(*id*) JEOD_MAKE_MESSAGE_CODE(TemplateMessages, "utils/model_template/", id)

The goal in a model message code implementation file is to create storage for and initialize each of static data members declared in the model message class.

There are several ways to do this. The examples below target the fictitious message class FooMessages defined in the model utils/foo. The class defines two message codes, FooMessages::fatal and FooMessages::warning.

- Most primitive:

```
const char * FooMessages::fatal = "utils/foo/fatal";
const char * FooMessages::warning = "utils/foo/warning";
```

- Recognizing that "utils/foo/" is common:

```
#define PATH "utils/foo/"
const char * FooMessages::fatal = PATH "fatal";
const char * FooMessages::warning = PATH "warning";
```

- Using the JEOD_MAKE_MESSAGE_CODE macro:

```
#define PATH "utils/foo/"
JEOD_MAKE_MESSAGE_CODE (FooMessages, PATH, fatal);
JEOD_MAKE_MESSAGE_CODE (FooMessages, PATH, warning);
```

- Using the model-specific macro MAKE_FOO_MESSAGE_CODE:

```
#define MAKE_FOO_MESSAGE_CODE(id) \
   JEOD_MAKE_MESSAGE_CODE(FooMessages, "utils/foo/", id)
MAKE_FOO_MESSAGE_CODE (fatal);
MAKE_FOO_MESSAGE_CODE (warning);
```

Define TemplateMessages data member *id*.

Parameters

id	The name of the data member.

Definition at line 86 of file template_messages.cc.

8.4 template_messages.hh File Reference

Define the class TemplateMessages, the class that specifies the message IDs used in the model template model.

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

Data Structures

- class jeod::TemplateMessages

  Declares messages associated with the model template model.

Namespaces

- jeod

  Namespace jeod.

8.4.1 Detailed Description

Define the class TemplateMessages, the class that specifies the message IDs used in the model template model.

Definition in file template_messages.hh.

Index