C++ UTC Time Library

Generated by Doxygen 1.8.1.2

Wed Aug 28 2013 14:03:14

Contents

Chapter 1

Todo List

Member utctime::validate_date (const int year, const int month, const int day, const int hour, const int minute, const int second)

Why does this throw an exception? Why not return false? In order so that the error message can show why it's invalid?

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 3

Class Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 utctime Namespace Reference

Namespace for UTCTime() class and associated functions.

Classes

- class UTCTimeException

  Base exception class.
- class bad_time_init

  Thrown when the time cannot be initialized.
- class bad_time

  Thrown when the time cannot be obtained.
- class invalid_date

  Thrown when an invalid date is provided to the constructor.
- class UTCTime

Functions

- bool validate_date (const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks whether a supplied date is valid.
- time_t get_day_diff ()

  Returns a time_t interval representing one day.
- time_t get_hour_diff ()

  Returns a time_t interval representing one hour.
- time_t get_sec_diff ()

  Returns a time_t interval representing one second.
- int tm_compare (const std::tm &first, const std::tm &second)

  Compares two std::tm structs.
- int tm_intraday_secs_diff (const std::tm &first, const std::tm &second)

  Returns the difference between two std::tm structs.
- bool is_leap_year (const int year)

  Checks if the supplied year is a leap year.
- std::tm * tm_increment_day (std::tm *changing_tm, const int quantity=1)

  Adds one or more days to a std::tm struct.
- std::tm * tm_increment_hour (std::tm *changing_tm, const int quantity=1)

Adds one or more hours to a std::tm struct.

- std::tm * tm_increment_minute (std::tm *changing_tm, const int quantity=1)

  Adds one or more minutes to a std::tm struct.
- std::tm * tm_increment_second (std::tm *changing_tm, const int quantity=1)

  Adds one or more seconds to a std::tm struct.
- std::tm * tm_decrement_day (std::tm *changing_tm, const int quantity=1)

  Deducts one or more days from a std::tm struct.
- std::tm * tm_decrement_hour (std::tm *changing_tm, const int quantity=1)

  Deducts one or more hours from a std::tm struct.
- std::tm * tm_decrement_minute (std::tm *changing_tm, const int quantity=1)

  Deducts one or more minutes from a std::tm struct.
- std::tm * tm_decrement_second (std::tm *changing_tm, const int quantity=1)

  Deducts one or more seconds from a std::tm struct.
- bool check_utc_timestamp (const time_t check_time, int &secs_diff, const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks if a UTC timestamp is accurate.
- time_t get_utc_timestamp (const int year, const int month, const int day, const int hour, const int minute, const int second)

  Gets a time_t timestamp for a requested UTC time.
- int get_utc_timestamp_sec_diff (const time_t check_time, const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks a time_t timestamp against a UTC time.

6.1.1 Function Documentation

6.1.1.1 bool utctime::check_utc_timestamp (const time_t *check_time*, int & *secs_diff*, const int *year*, const int *month*, const int *day*, const int *hour*, const int *minute*, const int *second*)

Checks if a UTC timestampe is accurate. A time_t timestamp is computed from the supplied datetime elements, and compared to the supplied time_t timestamp. The difference between the two, in seconds, is stored in the supplied secs_diff argument. This function is needed because the methodology used to calculate a timestamp by this library can sometimes be inaccurate when leap seconds or other unpredictable calendar changes occur. We therefore need a method to check if the returned timestamp is accurate. Other functions provided in this library call this function, so the user should not normally need to call it.

Parameters

check_time	The time_t timestamp to check
secs_diff	Modified to contain the difference, in seconds
year	The year
month	The month, 1 to 12
day	The day, 1 to 31, depending on the month
hour	The hour, 0 to 23
minute	The minute, 0 to 59
second	The second, 0 to 59

Returns

true if the supplied timestamp is accurate, false otherwise

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.2 time_t utctime::get_day_diff ( )

Returns a time_t interval representing one day. The C standard does not define the units in which a time_t value is measured. On POSIX-compliant systems it is measured in seconds, but we cannot assume this for full portability.

Returns

A time_t interval representing one day.

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.3 time_t utctime::get_hour_diff ( )

Returns a time_t interval representing one hour. The C standard does not define the units in which a time_t value is measured. On POSIX-compliant systems it is measured in seconds, but we cannot assume this for full portability.

Returns

A time_t interval representing one hour.

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.4 time_t utctime::get_sec_diff ( )

Returns a time_t interval representing one second. The C standard does not define the units in which a time_t value is measured. On POSIX-compliant systems it is measured in seconds, but we cannot assume this for full portability.

Returns

A time_t interval representing one second.

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.5 time_t utctime::get_utc_timestamp (const int *year*, const int *month*, const int *day*, const int *hour*, const int *minute*, const int *second*)

Gets a time_t timestamp for a requested UTC time.

Parameters

year	The year
month	The month, 1 to 12
day	The day, 1 to 31, depending on the month
hour	The hour, 0 to 23
minute	The minute, 0 to 59
second	The second, 0 to 59. Leap seconds are not supported.

Returns

A time_t timestamp for the requested UTC time.

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.6 int utctime::get_utc_timestamp_sec_diff (const time_t *check_time,* const int *year,* const int *month,* const int *day,* const int *hour,* const int *minute,* const int *second*)

Checks a time_t timestamp against a UTC time, and returns the difference in seconds. This function only returns a good value if the timestamp is less than 24 hours away from the desired time, so the caller is responsible for making sure that it is. This function may also return a bad value if a leap second or other unpredictable calendar change falls between the desired UTC time and the provided time stamp. The result should therefore always be checked with utctime::check_utc_timestamp(), or by calling this function again.

Parameters

check_time	The time_t timestamp to check
year	The year
month	The month, 1 to 12
day	The day, 1 to 31, depending on the month
hour	The hour, 0 to 23
minute	The minute, 0 to 59
second	The second, 0 to 59. Leap seconds are not supported.

Returns

The difference, if any, represented in seconds.

Exceptions

bad_time	if the current time cannot be obtained.

6.1.1.7 bool utctime::is_leap_year (const int *year*)

Checks if the supplied year is a leap year.

Parameters

year	A year

Returns

`true` is `year` is a leap year, `false` otherwise.

6.1.1.8 int utctime::tm_compare (const std::tm & *first,* const std::tm & *second*)

Compares two std::tm structs. Only the year, month, day, hour, minute and second are compared. Any timezone or DST information is ignored.

Parameters

first	The first std::tm struct.
second	The second std::tm struct.

Returns

-1 if `first` is earlier than `second`, 1 if `first` is later than `second`, and 0 if `first` is equal to `second`.

6.1.1.9 std::tm * utctime::tm_decrement_day (std::tm * *changing_tm*, const int *quantity* = `1`)

Deducts one or more days from a std::tm struct, decrementing the month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of days to deduct

Returns

A pointer to the same std::tm struct.

6.1.1.10 std::tm * utctime::tm_decrement_hour (std::tm * *changing_tm*, const int *quantity* = `1`)

Deducts one or more hours from a std::tm struct, decrementing the day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of hours to deduct

Returns

A pointer to the same std::tm struct.

6.1.1.11 std::tm * utctime::tm_decrement_minute (std::tm * *changing_tm*, const int *quantity* = `1`)

Deducts one or more minutes from a std::tm struct, decrementing the hour, day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of minutes to deduct

Returns

A pointer to the same std::tm struct.

6.1.1.12 std::tm * utctime::tm_decrement_second (std::tm * *changing_tm,* const int *quantity* = 1)

Deducts one or more seconds from a std::tm struct, decrementing the minute, hour, day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of seconds to deduct

Returns

A pointer to the same std::tm struct.

6.1.1.13 std::tm * utctime::tm_increment_day (std::tm * *changing_tm,* const int *quantity* = 1)

Adds one or more days to a std::tm struct, incrementing the month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of days to add

Returns

A pointer to the same std::tm struct.

6.1.1.14 std::tm * utctime::tm_increment_hour (std::tm * *changing_tm,* const int *quantity* = 1)

Adds one or more hours to a std::tm struct, incrementing the day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of hours to add

Returns

A pointer to the same std::tm struct.

6.1.1.15 std::tm * utctime::tm_increment_minute (std::tm * *changing_tm,* const int *quantity* = 1)

Adds one or more minutes to a std::tm struct, incrementing the hour, day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of minutes to add

Returns

A pointer to the same std::tm struct.

6.1.1.16 std::tm * utctime::tm_increment_second (std::tm * *changing_tm,* const int *quantity* = `1`)

Adds one or mor seconds to a std::tm struct, incrementing the minute, hour, day, month and/or the year as necessary.

Parameters

changing_tm	A pointer to the std::tm struct to increment. The struct referred to by the pointer is modified by the function.
quantity	The number of seconds to add

Returns

A pointer to the same std::tm struct.

6.1.1.17 int utctime::tm_intraday_secs_diff (const std::tm & *first,* const std::tm & *second*)

Returns the difference between two std::tm structs. The structs are assumed to be within 24 hours of each other, and if they are not, the returned result is computed as if they were. For instance, comparing 10:00 on one day to 14:00 on the next day will yield a difference equivalent to 4 hours, not to 28 hours.

Parameters

first	The first std::tm struct
second	The second std::tm struct

Returns

The difference, in seconds, between the two std::tm structs. The difference is positive if `first` is earlier than `second`, and negative if `second` is earlier than `first`.

6.1.1.18 bool utctime::validate_date (const int *year,* const int *month,* const int *day,* const int *hour,* const int *minute,* const int *second*)

Checks whether a supplied date is valid.

Parameters

year	The year
month	The month, 1 to 12
day	The day, 1 to 31, depending on the month
hour	The hour, 0 to 23
minute	The minute, 0 to 59
second	The second, 0 to 59. Leap seconds are not supported.

Returns

true if the date if valid.

Exceptions

invalid_date	if the date is not valid.

Todo Why does this throw an exception? Why not return false? In order so that the error message can show why it's invalid?

Chapter 7

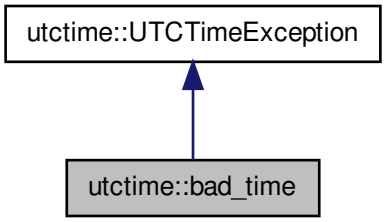
Class Documentation

7.1 utctime::bad_time Class Reference

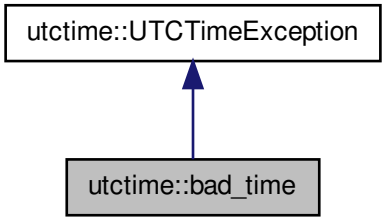
Thrown when the time cannot be obtained.

`#include <utctime.h>`

Inheritance diagram for utctime::bad_time:



Collaboration diagram for utctime::bad_time:



Public Member Functions

- bad_time ()
  Constructor.

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

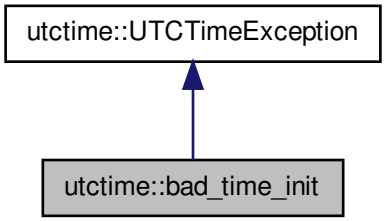
- utctime.h

7.2 utctime::bad_time_init Class Reference

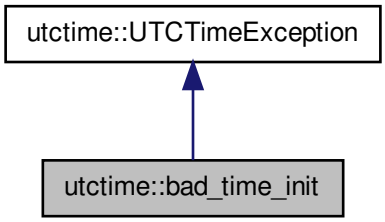
Thrown when the time cannot be initialized.

`#include <utctime.h>`

Inheritance diagram for utctime::bad_time_init:



Collaboration diagram for utctime::bad_time_init:



Public Member Functions

- bad_time_init ()
  Constructor.

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

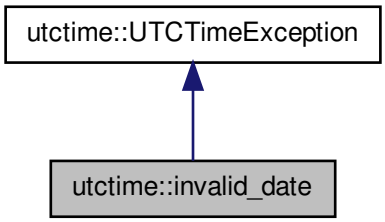
- utctime.h

7.3 utctime::invalid_date Class Reference

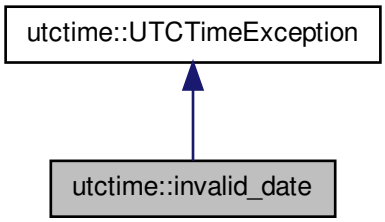
Thrown when an invalid date is provided to the constructor.

`#include <utctime.h>`

Inheritance diagram for utctime::invalid_date:



Collaboration diagram for utctime::invalid_date:



Public Member Functions

- invalid_date (const std::string &msg)

  Constructor.

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

- utctime.h

7.4 utctime::UTCTime Class Reference

`#include <utctime.h>`

Public Member Functions

- UTCTime ()

  Default constructor.
- UTCTime (const std::tm &utc_tm)

  Constructor taking a std::tm struct.
- UTCTime (const int year, const int month, const int day, const int hour, const int minute, const int second)

  Constructor taking individual date values.
- std::tm get_tm () const

  Returns a std::tm struct containing the UTC datetime.
- std::string time_string () const

  Returns a std::string containing the UTC datetime.
- std::string time_string_inet () const

  Returns a std::string containing the UTC datetime in RFC3339 format.
- time_t timestamp () const

  Returns a time_t timestamp for the UTC datetime.
- bool operator< (const UTCTime &rhs) const

  Overloaded less than operator.
- bool operator>= (const UTCTime &rhs) const

  Overloaded greater than or equal to operator.
- bool operator> (const UTCTime &rhs) const

  Overloaded greater than operator.
- bool operator<= (const UTCTime &rhs) const

  Overloaded less than or equal to operator.
- bool operator== (const UTCTime &rhs) const

  Overloaded equality operator.
- bool operator!= (const UTCTime &rhs) const

  Overloaded inequality operator.
- double operator- (const UTCTime &rhs) const

  Overloaded subtraction operator.

7.4.1 Detailed Description

A class for holding a UTC time.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 UTCTime::UTCTime ( ) [explicit]

Default constructor, which initializes to the current time.

Exceptions

bad_time_init	if unable to get the current time.

7.4.2.2 UTCTime::UTCTime (const std::tm & *utc_tm*) [explicit]

Constructor taking a std::tm struct.

Parameters

utc_tm	A std::tm struct containing the desired initialization time.

Exceptions

bad_time_init	if unable to get the current time.
invalid_date	if the supplied date is bad.

7.4.2.3 UTCTime::UTCTime (const int *year,* const int *month,* const int *day,* const int *hour,* const int *minute,* const int *second*) [explicit]

Constructor taking individual date values.

Parameters

year	The year
month	The month, 1 to 12
day	The day, 1 to 31, depending on month
hour	The hour, 0 to 23
minute	The minute, 0 to 59
second	The second, 0 to 59. Leap seconds are not supported.

Exceptions

bad_time_init	if unable to get the current time.
invalid_date	if the supplied date is bad.

7.4.3 Member Function Documentation

7.4.3.1 std::tm UTCTime::get_tm ( ) const

Returns a std::tm struct containing the UTC datetime.

Returns

A std::tm struct containing the UTC datetime.

7.4.3.2 bool UTCTime::operator!= (const UTCTime & *rhs*) const

Overloaded inequality operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is not equal to `rhs`, false otherwise.

7.4.3.3 double UTCTime::operator- (const UTCTime & *rhs*) const

Overloaded subtraction operator

Parameters

rhs	The UTCTime instance to subtract from the instance

Returns

The difference, in seconds, between the two instances.

7.4.3.4 bool UTCTime::operator< (const UTCTime & *rhs*) const

Overloaded less than operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is less than `rhs`, false otherwise.

7.4.3.5 bool UTCTime::operator<= (const UTCTime & *rhs*) const

Overloaded less than or equal to operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is less than or equal to `rhs`, false otherwise.

7.4.3.6 bool UTCTime::operator== (const UTCTime & *rhs*) const

Overloaded equality operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is equal to `rhs`, false otherwise.

7.4.3.7 bool UTCTime::operator> (const UTCTime & *rhs*) const

Overloaded greater than operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is greater than `rhs`, false otherwise.

7.4.3.8 bool UTCTime::operator>= (const UTCTime & *rhs*) const

Overloaded greater than or equal to operator

Parameters

rhs	The UTCTime instance to which to compare

Returns

true if the instance is greater than or equal to `rhs`, false otherwise.

7.4.3.9 std::string UTCTime::time_string ( ) const

Returns a std::string containing the UTC datetime.

Returns

A std::string containing the UTC datetime.

Exceptions

bad_time	if unable to get the time.

7.4.3.10 std::string UTCTime::time_string_inet ( ) const

Returns a std::string containing the UTC datetime in RFC3339 format.

Returns

A std::string containing the UTC datetime in RFC3339 format.

Exceptions

bad_time	if unable to get the time.

7.4.3.11 time_t UTCTime::timestamp ( ) const

Returns a time_t timestamp for the UTC datetime.

Returns

A time_t timestamp for the UTC datetime.

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

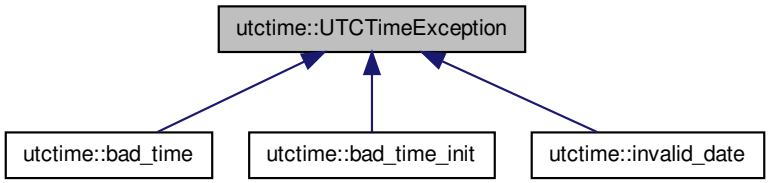
- utctime.h
- utctime.cpp

7.5 utctime::UTCTimeException Class Reference

Base exception class.

`#include <utctime.h>`

Inheritance diagram for utctime::UTCTimeException:



Public Member Functions

- UTCTimeException (const std::string msg="No error message")

  Constructor.
- virtual ~UTCTimeException ()

  Destructor.
- const std::string & what () const

  Returns an error message.

7.5.1 Detailed Description

All custom exceptions thrown by the UTCTime() class and associated functions are derived from UTCTimeException.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 utctime::UTCTimeException::UTCTimeException (const std::string *msg* = "No error message") [inline], [explicit]

Parameters

msg	An error message to print when `what()` is called.

7.5.3 Member Function Documentation

7.5.3.1 const std::string& utctime::UTCTimeException::what ( ) const [inline]

Returns

A std::string containing the error message.

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

- utctime.h

Chapter 8

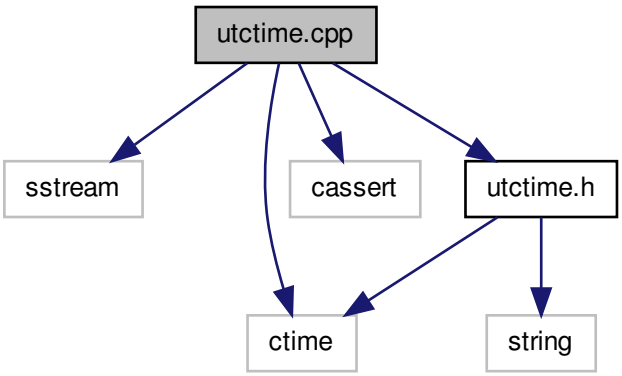
File Documentation

8.1 utctime.cpp File Reference

Implementation of UTCTime() class and associated functions.

```
#include <sstream>
#include <ctime>
#include <cassert>
#include "utctime.h"
```

Include dependency graph for utctime.cpp:



8.1.1 Detailed Description

Implementation of UTCTime() class and associated functions.

Author

Paul Griffiths

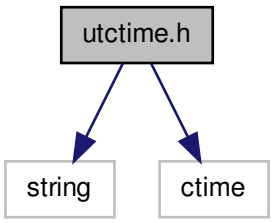
Copyright

8.2 utctime.h File Reference

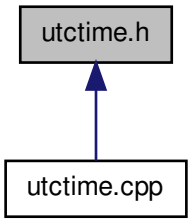
Interface to UTCTime() class and associated functions.

```
#include <string>
#include <ctime>
```

Include dependency graph for utctime.h:



This graph shows which files directly or indirectly include this file:



Classes

- class utctime::UTCTimeException

  Base exception class.
- class utctime::bad_time_init

  Thrown when the time cannot be initialized.
- class utctime::bad_time

  Thrown when the time cannot be obtained.
- class utctime::invalid_date

  Thrown when an invalid date is provided to the constructor.
- class utctime::UTCTime

Namespaces

- namespace utctime

Namespace for UTCTime() class and associated functions.

Functions

- bool utctime::validate_date (const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks whether a supplied date is valid.
- time_t utctime::get_day_diff ()

  Returns a time_t interval representing one day.
- time_t utctime::get_hour_diff ()

  Returns a time_t interval representing one hour.
- time_t utctime::get_sec_diff ()

  Returns a time_t interval representing one second.
- int utctime::tm_compare (const std::tm &first, const std::tm &second)

  Compares two std::tm structs.
- int utctime::tm_intraday_secs_diff (const std::tm &first, const std::tm &second)

  Returns the difference between two std::tm structs.
- bool utctime::is_leap_year (const int year)

  Checks if the supplied year is a leap year.
- std::tm * utctime::tm_increment_day (std::tm *changing_tm, const int quantity=1)

  Adds one or more days to a std::tm struct.
- std::tm * utctime::tm_increment_hour (std::tm *changing_tm, const int quantity=1)

  Adds one or more hours to a std::tm struct.
- std::tm * utctime::tm_increment_minute (std::tm *changing_tm, const int quantity=1)

  Adds one or more minutes to a std::tm struct.
- std::tm * utctime::tm_increment_second (std::tm *changing_tm, const int quantity=1)

  Adds one or more seconds to a std::tm struct.
- std::tm * utctime::tm_decrement_day (std::tm *changing_tm, const int quantity=1)

  Deducts one or more days from a std::tm struct.
- std::tm * utctime::tm_decrement_hour (std::tm *changing_tm, const int quantity=1)

  Deducts one or more hours from a std::tm struct.
- std::tm * utctime::tm_decrement_minute (std::tm *changing_tm, const int quantity=1)

  Deducts one or more minutes from a std::tm struct.
- std::tm * utctime::tm_decrement_second (std::tm *changing_tm, const int quantity=1)

  Deducts one or more seconds from a std::tm struct.
- bool utctime::check_utc_timestamp (const time_t check_time, int &secs_diff, const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks if a UTC timestamp is accurate.
- time_t utctime::get_utc_timestamp (const int year, const int month, const int day, const int hour, const int minute, const int second)

  Gets a time_t timestamp for a requested UTC time.
- int utctime::get_utc_timestamp_sec_diff (const time_t check_time, const int year, const int month, const int day, const int hour, const int minute, const int second)

  Checks a time_t timestamp against a UTC time.

8.2.1 Detailed Description

Interface to UTCTime() class and associated functions.

Author

Paul Griffiths

Copyright

Index