

globus gass copy

5.4

Generated by Doxygen 1.5.6

Mon Apr 26 16:52:32 2010

# Contents

<b>1</b>	<b>Main Page</b>	<b>2</b>
<b>2</b>	<b>Data Structure Index</b>	<b>2</b>
2.1	Data Structures . . . . .	2
<b>3</b>	<b>File Index</b>	<b>2</b>
3.1	File List . . . . .	2
<b>4</b>	<b>Data Structure Documentation</b>	<b>2</b>
4.1	globus_gass_copy_glob_stat_t Struct Reference . . . . .	2
4.1.1	Detailed Description . . . . .	3
4.1.2	Field Documentation . . . . .	3
4.2	globus_gass_copy_state_s Struct Reference . . . . .	3
4.2.1	Detailed Description . . . . .	4
4.2.2	Field Documentation . . . . .	4
4.3	globus_i_gass_copy_buffer_t Struct Reference . . . . .	4
4.3.1	Detailed Description . . . . .	4
4.4	globus_i_gass_copy_cancel_s Struct Reference . . . . .	4
4.4.1	Detailed Description . . . . .	4
4.5	globus_i_gass_copy_monitor_t Struct Reference . . . . .	5
4.5.1	Detailed Description . . . . .	5
4.6	globus_i_gass_copy_state_target_s Struct Reference . . . . .	5
4.6.1	Detailed Description . . . . .	5
4.6.2	Field Documentation . . . . .	5
<b>5</b>	<b>File Documentation</b>	<b>7</b>
5.1	globus_gass_copy.c File Reference . . . . .	7
5.1.1	Detailed Description . . . . .	8
5.1.2	Function Documentation . . . . .	8
5.2	globus_gass_copy.h File Reference . . . . .	19
5.2.1	Detailed Description . . . . .	20
5.2.2	Define Documentation . . . . .	20
5.2.3	Typedef Documentation . . . . .	21
5.2.4	Enumeration Type Documentation . . . . .	21
5.2.5	Function Documentation . . . . .	21

# 1 Main Page

The Globus GASS Copy library is motivated by the desire to provide a uniform interface to transfer files specified by different protocols. The goals in doing this are to:

- Provide a robust way to describe and apply file transfer properties for a variety of protocols. These include the standard HTTP, FTP and GSIFTP options. Some of the new file transfer capabilities in GSIFTP are parallel, striping, authentication and TCP buffer sizing.
- Provide a service to support nonblocking file transfer and handle asynchronous file and network events.
- Provide a simple and portable way to implement file transfers.

Any program that uses Globus GASS Copy functions must include "globus\_gass\_copy.h".

## 2 Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">globus_gass_copy_glob_stat_t</a> (Glob expanded entry information )	2
<a href="#">globus_gass_copy_state_s</a> (The state structure contains all that is required to perform a file transfer from a source to a destination )	3
<a href="#">globus_i_gass_copy_buffer_t</a> (The buffer structure used for read/write queue entries )	4
<a href="#">globus_i_gass_copy_cancel_s</a> (Gass copy cancel struct )	4
<a href="#">globus_i_gass_copy_monitor_t</a> (The state monitor struct )	5
<a href="#">globus_i_gass_copy_state_target_s</a> (GASS copy target (e.g )	5

## 3 File Index

### 3.1 File List

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

<a href="#">globus_gass_copy.c</a> (Globus GASS Copy library )	7
<a href="#">globus_gass_copy.h</a> (Header file for the gass copy library )	19

## 4 Data Structure Documentation

### 4.1 globus\_gass\_copy\_glob\_stat\_t Struct Reference

Glob expanded entry information.

## Data Fields

- `globus_gass_copy_glob_entry_t` type
- `char * unique_id`
- `char * symlink_target`
- `int mode`
- `int mdtm`
- `globus_off_t size`

### 4.1.1 Detailed Description

Glob expanded entry information.

### 4.1.2 Field Documentation

#### 4.1.2.1 `globus_gass_copy_glob_entry_t globus_gass_copy_glob_stat_t::type`

The file type of the entry.

#### 4.1.2.2 `char* globus_gass_copy_glob_stat_t::unique_id`

A string that uniquely identifies the data that the entry refers to.

A file and a symlink to that file will have the same `unique_id`. It is NULL for when not available.

#### 4.1.2.3 `char* globus_gass_copy_glob_stat_t::symlink_target`

This points to the full path of the target of a symlink.

It is NULL for non-symlinks or when not available.

#### 4.1.2.4 `int globus_gass_copy_glob_stat_t::mode`

An integer specifying the mode of the file.

It is set to -1 when not available.

#### 4.1.2.5 `int globus_gass_copy_glob_stat_t::mdtm`

An integer specifying the modification time of the file.

It is set to -1 when not available.

#### 4.1.2.6 `globus_off_t globus_gass_copy_glob_stat_t::size`

A `globus_off_t` specifying the size of the file.

It is set to -1 when not available.

## 4.2 `globus_gass_copy_state_s` Struct Reference

The state structure contains all that is required to perform a file transfer from a source to a destination.

## Data Fields

- [globus\\_i\\_gass\\_copy\\_target\\_t](#) `source`
- [globus\\_i\\_gass\\_copy\\_target\\_t](#) `dest`
- [globus\\_i\\_gass\\_copy\\_monitor\\_t](#) `monitor`
- [globus\\_mutex\\_t](#) `mutex`
- [globus\\_i\\_gass\\_copy\\_cancel\\_status\\_t](#) `cancel`

### 4.2.1 Detailed Description

The state structure contains all that is required to perform a file transfer from a source to a destination.

### 4.2.2 Field Documentation

#### 4.2.2.1 `globus_i_gass_copy_target_t globus_gass_copy_state_s::source`

Source information for the file transfer.

#### 4.2.2.2 `globus_i_gass_copy_target_t globus_gass_copy_state_s::dest`

Dest information for the file transfer.

#### 4.2.2.3 `globus_i_gass_copy_monitor_t globus_gass_copy_state_s::monitor`

Used for signalling from the various callback functions.

#### 4.2.2.4 `globus_mutex_t globus_gass_copy_state_s::mutex`

coordinates the modifying of the state, aside from the target structures

#### 4.2.2.5 `globus_i_gass_copy_cancel_status_t globus_gass_copy_state_s::cancel`

indicates the status of the cancel operation.

## 4.3 `globus_i_gass_copy_buffer_t` Struct Reference

The buffer structure used for read/write queue entries.

### 4.3.1 Detailed Description

The buffer structure used for read/write queue entries.

## 4.4 `globus_i_gass_copy_cancel_s` Struct Reference

gass copy cancel struct

### 4.4.1 Detailed Description

gass copy cancel struct

## 4.5 globus\_i\_gass\_copy\_monitor\_t Struct Reference

The state monitor struct.

### 4.5.1 Detailed Description

The state monitor struct.

## 4.6 globus\_i\_gass\_copy\_state\_target\_s Struct Reference

GASS copy target (e.g.

### Data Fields

- char \* [url](#)
- globus\_gass\_copy\_attr\_t \* [attr](#)
- globus\_mutex\_t [mutex](#)
- globus\_fifo\_t [queue](#)
- int [n\\_pending](#)
- int [n\\_simultaneous](#)
- int [n\\_complete](#)
- globus\_i\_gass\_copy\_target\_status\_t [status](#)
- globus\_gass\_copy\_url\_mode\_t [mode](#)
- union {
  - struct {
    - } [ftp](#)
  - struct {
    - globus\_gass\_transfer\_request\_t [request](#)
  - } [gass](#)
  - struct {
    - globus\_bool\_t [free\\_handle](#)
    - globus\_bool\_t [seekable](#)
  - } [io](#)
- } [data](#)

### 4.6.1 Detailed Description

GASS copy target (e.g.

source, destination) transfer information.

### 4.6.2 Field Documentation

#### 4.6.2.1 char\* globus\_i\_gass\_copy\_state\_target\_s::url

url for file transfer

#### 4.6.2.2 globus\_gass\_copy\_attr\_t\* globus\_i\_gass\_copy\_state\_target\_s::attr

attributes to control file transfer

#### **4.6.2.3 globus\_mutex\_t globus\_i\_gass\_copy\_state\_target\_s::mutex**

coordinates the modifying of the target structure

#### **4.6.2.4 globus\_fifo\_t globus\_i\_gass\_copy\_state\_target\_s::queue**

a queue to manage the reading/writing of data buffers

#### **4.6.2.5 int globus\_i\_gass\_copy\_state\_target\_s::n\_pending**

Used for keeping track of reads/writes in the read/write queue.

#### **4.6.2.6 int globus\_i\_gass\_copy\_state\_target\_s::n\_simultaneous**

Used to limit the number of n\_pending.

#### **4.6.2.7 int globus\_i\_gass\_copy\_state\_target\_s::n\_complete**

Used to compute the offset for ftp writes.

#### **4.6.2.8 globus\_i\_gass\_copy\_target\_status\_t globus\_i\_gass\_copy\_state\_target\_s::status**

signifies the target has been successfully setup

#### **4.6.2.9 globus\_gass\_copy\_url\_mode\_t globus\_i\_gass\_copy\_state\_target\_s::mode**

mode used to identify the below target union struct.

#### **4.6.2.10 struct { ... } globus\_i\_gass\_copy\_state\_target\_s::ftp**

ftp specific data

#### **4.6.2.11 globus\_gass\_transfer\_request\_t globus\_i\_gass\_copy\_state\_target\_s::request**

GASS equivalent of a handle.

#### **4.6.2.12 struct { ... } globus\_i\_gass\_copy\_state\_target\_s::gass**

GASS specific data.

#### **4.6.2.13 globus\_bool\_t globus\_i\_gass\_copy\_state\_target\_s::free\_handle**

If the IO handle was passed as an argument then FALSE If the IO handle was created internally then TRUE.

#### **4.6.2.14 globus\_bool\_t globus\_i\_gass\_copy\_state\_target\_s::seekable**

Can globus\_io\_file\_seek() be performed on this handle?

#### **4.6.2.15 struct { ... } globus\_i\_gass\_copy\_state\_target\_s::io**

IO specific data.

#### **4.6.2.16 union { ... } globus\_i\_gass\_copy\_state\_target\_s::data**

data required to perform each type of transfer

## 5 File Documentation

### 5.1 globus\_gass\_copy.c File Reference

Globus GASS Copy library.

#### Functions

- globus\_result\_t [globus\\_gass\\_copy\\_handle\\_init](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_gass\_copy\_handleattr\_t \*attr)
- globus\_result\_t [globus\\_gass\\_copy\\_handle\\_destroy](#) (globus\_gass\_copy\_handle\_t \*handle)
- globus\_result\_t [globus\\_gass\\_copy\\_set\\_buffer\\_length](#) (globus\_gass\_copy\_handle\_t \*handle, int length)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_buffer\\_length](#) (globus\_gass\_copy\_handle\_t \*handle, int \*length)
- globus\_result\_t [globus\\_gass\\_copy\\_set\\_no\\_third\\_party\\_transfers](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_bool\_t no\_third\_party\_transfers)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_no\\_third\\_party\\_transfers](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_bool\_t \*no\_third\_party\_transfers)
- globus\_result\_t [globus\\_gass\\_copy\\_set\\_allocate](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_bool\_t send\_allo)
- globus\_result\_t [globus\\_gass\\_copy\\_set\\_partial\\_offsets](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_off\_t offset, globus\_off\_t end\_offset)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_partial\\_offsets](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_off\_t \*offset, globus\_off\_t \*end\_offset)
- globus\_result\_t [globus\\_gass\\_copy\\_attr\\_init](#) (globus\_gass\_copy\_attr\_t \*attr)
- globus\_result\_t [globus\\_gass\\_copy\\_attr\\_set\\_ftp](#) (globus\_gass\_copy\_attr\_t \*attr, globus\_ftp\_client\_operationattr\_t \*ftp\_attr)
- globus\_result\_t [globus\\_gass\\_copy\\_attr\\_set\\_io](#) (globus\_gass\_copy\_attr\_t \*attr, globus\_io\_attr\_t \*io\_attr)
- globus\_result\_t [globus\\_gass\\_copy\\_attr\\_set\\_gass](#) (globus\_gass\_copy\_attr\_t \*attr, globus\_gass\_transfer\_requestattr\_t \*gass\_attr)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_url\\_mode](#) (char \*url, globus\_gass\_copy\_url\_mode\_t \*mode)
- globus\_result\_t [globus\\_gass\\_copy\\_register\\_performance\\_cb](#) (globus\_gass\_copy\_handle\_t \*handle, [globus\\_gass\\_copy\\_performance\\_cb\\_t](#) callback, void \*user\_arg)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_status](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_gass\_copy\_status\_t \*status)
- const char \* [globus\\_gass\\_copy\\_get\\_status\\_string](#) (globus\_gass\_copy\_handle\_t \*handle)
- globus\_result\_t [globus\\_gass\\_copy\\_url\\_to\\_url](#) (globus\_gass\_copy\_handle\_t \*handle, char \*source\_url, globus\_gass\_copy\_attr\_t \*source\_attr, char \*dest\_url, globus\_gass\_copy\_attr\_t \*dest\_attr)
- globus\_result\_t [globus\\_gass\\_copy\\_url\\_to\\_handle](#) (globus\_gass\_copy\_handle\_t \*handle, char \*source\_url, globus\_gass\_copy\_attr\_t \*source\_attr, globus\_io\_handle\_t \*dest\_handle)
- globus\_result\_t [globus\\_gass\\_copy\\_handle\\_to\\_url](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_io\_handle\_t \*source\_handle, char \*dest\_url, globus\_gass\_copy\_attr\_t \*dest\_attr)
- globus\_result\_t [globus\\_gass\\_copy\\_register\\_url\\_to\\_url](#) (globus\_gass\_copy\_handle\_t \*handle, char \*source\_url, globus\_gass\_copy\_attr\_t \*source\_attr, char \*dest\_url, globus\_gass\_copy\_attr\_t \*dest\_attr, globus\_gass\_copy\_callback\_t callback\_func, void \*callback\_arg)
- globus\_result\_t [globus\\_gass\\_copy\\_register\\_url\\_to\\_handle](#) (globus\_gass\_copy\_handle\_t \*handle, char \*source\_url, globus\_gass\_copy\_attr\_t \*source\_attr, globus\_io\_handle\_t \*dest\_handle, globus\_gass\_copy\_callback\_t callback\_func, void \*callback\_arg)
- globus\_result\_t [globus\\_gass\\_copy\\_register\\_handle\\_to\\_url](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_io\_handle\_t \*source\_handle, char \*dest\_url, globus\_gass\_copy\_attr\_t \*dest\_attr, globus\_gass\_copy\_callback\_t callback\_func, void \*callback\_arg)
- globus\_result\_t [globus\\_gass\\_copy\\_cache\\_url\\_state](#) (globus\_gass\_copy\_handle\_t \*handle, char \*url)
- globus\_result\_t [globus\\_gass\\_copy\\_flush\\_url\\_state](#) (globus\_gass\_copy\_handle\_t \*handle, char \*url)



- globus\_result\_t [globus\\_gass\\_copy\\_set\\_user\\_pointer](#) (globus\_gass\_copy\_handle\_t \*handle, void \*user\_pointer)
- globus\_result\_t [globus\\_gass\\_copy\\_get\\_user\\_pointer](#) (globus\_gass\_copy\_handle\_t \*handle, void \*\*user\_data)
- globus\_result\_t [globus\\_gass\\_copy\\_cancel](#) (globus\_gass\_copy\_handle\_t \*handle, globus\_gass\_copy\_callback\_t cancel\_callback, void \*cancel\_callback\_arg)
- globus\_result\_t [globus\\_l\\_gass\\_copy\\_target\\_cancel](#) (globus\_i\_gass\_copy\_cancel\_t \*cancel\_info)

### 5.1.1 Detailed Description

Globus GASS Copy library.

#### See also:

See the detailed description in [globus\\_gass\\_copy.h](#)

### 5.1.2 Function Documentation

#### 5.1.2.1 globus\_result\_t globus\_gass\_copy\_handle\_init (globus\_gass\_copy\_handle\_t \* *handle*, globus\_gass\_copy\_handleattr\_t \* *attr*)

Initialize a GASS Copy handle.

A globus\_gass\_copy\_handle must be initialized before any transfers may be associated with it. This function initializes a globus\_gass\_copy\_handle to be used for doing transfers, this includes initializing a globus\_ftp\_client\_handle which will be used for doing any ftp/gsftp transfers. The same handle may be used to perform multiple, consecutive transfers. However, there can only be one transfer associated with a particular handle at any given time. After all transfers to be associated with this handle have completed, the handle should be destroyed by calling [globus\\_gass\\_copy\\_handle\\_destroy\(\)](#).

#### Parameters:

*handle* The handle to be initialized

*attr* The handle attributes used to use with this handle

#### Returns:

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

#### See also:

[globus\\_gass\\_copy\\_handle\\_destroy\(\)](#) , [globus\\_gass\\_copy\\_handleattr\\_init\(\)](#), [globus\\_ftp\\_client\\_handle\\_init\(\)](#)

References GLOBUS\_GASS\_COPY\_MODULE.

#### 5.1.2.2 globus\_result\_t globus\_gass\_copy\_handle\_destroy (globus\_gass\_copy\_handle\_t \* *handle*)

Destroy a GASS Copy handle.

Destroy a gass\_copy\_handle, which was initialized using [globus\\_gass\\_copy\\_handle\\_init\(\)](#), that will no longer be used for doing transfers. Once the handle is destroyed, no further transfers should be associated with it.

#### Parameters:

*handle* The handle to be destroyed

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

**See also:**

[globus\\_gass\\_copy\\_handle\\_init\(\)](#), [globus\\_ftp\\_client\\_handle\\_destroy\(\)](#)

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.3 globus\_result\_t globus\_gass\_copy\_set\_buffer\_length (globus\_gass\_copy\_handle\_t \* *handle*, int *length*)**

Set the size of the buffer to be used for doing transfers.

This function allows the user to set the size of the buffer that will be used for doing transfers, if this function is not called the buffer size will default to 1M.

**Parameters:**

***handle*** Set the buffer length for transfers associated with this handle.

***length*** The length, in bytes, to make the buffer.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.4 globus\_result\_t globus\_gass\_copy\_get\_buffer\_length (globus\_gass\_copy\_handle\_t \* *handle*, int \* *length*)**

Get the size of the buffer being used for doing transfers.

This function allows the user to get the size of the buffer that is being used for doing transfers.

**Parameters:**

***handle*** Get the buffer length for transfers associated with this handle.

***length*** The length, in bytes, of the buffer.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.5 globus\_result\_t globus\_gass\_copy\_set\_no\_third\_party\_transfers (globus\_gass\_copy\_handle\_t \* *handle*, globus\_bool\_t *no\_third\_party\_transfers*)**

Turn third-party transfers on or off.

(They are on by default.)

This function allows the user to turn third-party transfers on or off for ftp to ftp transfers associated with a particular handle. This is often desired if one of the servers involved in the transfer does not allow third-party transfers.

**Parameters:**

*handle* Turn third-party transfers on or off for transfers associated with this handle. They are on by default.

*no\_third\_party\_transfers* GLOBUS\_FALSE if third-party transfers should be used. GLOBUS\_TRUE if third-party transfers should not be used.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References globus\_gass\_copy\_get\_status(), and GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.6 globus\_result\_t globus\_gass\_copy\_get\_no\_third\_party\_transfers (globus\_gass\_copy\_handle\_t \* handle, globus\_bool\_t \* no\_third\_party\_transfers)**

See if third-party transfers are turned on or off.

(They are on by default.)

This function allows the user to see if third-party transfers are turned on or off for ftp to ftp transfers associated with a particular handle. This is often desired if one of the servers involved in the transfer does not allow third-party transfers.

**Parameters:**

*handle* See if third-party transfers are turned on or off for transfers associated with this handle. They are on by default.

*no\_third\_party\_transfers* GLOBUS\_FALSE if third-party transfers should be used. GLOBUS\_TRUE if third-party transfers should not be used.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.7 globus\_result\_t globus\_gass\_copy\_set\_allocate (globus\_gass\_copy\_handle\_t \* handle, globus\_bool\_t send\_allo)**

Set allo on or off.

**5.1.2.8 globus\_result\_t globus\_gass\_copy\_set\_partial\_offsets (globus\_gass\_copy\_handle\_t \* handle, globus\_off\_t offset, globus\_off\_t end\_offset)**

Set the offsets to be used for doing partial transfers.

This function allows the user to set the offsets that will be used for doing partial transfers. An offset of -1 will disable partial transfers. An end\_offset of -1 means EOF.

**Parameters:**

*handle* Set the offsets for partial transfers associated with this handle.

*offset* The starting offset for the partial transfer.

*end\_offset* The ending offset for the partial transfer.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.9 globus\_result\_t globus\_gass\_copy\_get\_partial\_offsets (globus\_gass\_copy\_handle\_t \* *handle*, globus\_off\_t \* *offset*, globus\_off\_t \* *end\_offset*)**

Get the offsets being used for doing partial transfers.

This function allows the user to get the offsets that are being used for doing partial transfers. An offset of -1 means partial transfers are disabled.

**Parameters:**

*handle* Get the offsets for partial transfers associated with this handle.

*offset* The starting offset for the partial transfer.

*end\_offset* The ending offset for the partial transfer.

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.10 globus\_result\_t globus\_gass\_copy\_attr\_init (globus\_gass\_copy\_attr\_t \* *attr*)**

Initialize an attribute structure.

The globus\_gass\_copy\_attr\_t can be used to pass the globus\_gass\_copy library information about how a transfer should be performed. It must first be initialized by calling this function. Then any or all of the following functions may be called to set attributes associated with a particular protocol: [globus\\_gass\\_copy\\_attr\\_set\\_ftp\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_gass\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_io\(\)](#). Any function which takes a globus\_gass\_copy\_attr\_t as an argument will also accept GLOBUS\_NULL, in which case the appropriate set of default attributes will be used.

**Parameters:**

*attr* The attribute structure to be initialized

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

**See also:**

[globus\\_gass\\_copy\\_attr\\_set\\_ftp\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_gass\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_io\(\)](#), [globus\\_gass\\_copy\\_get\\_url\\_mode\(\)](#).

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.11 globus\_result\_t globus\_gass\_copy\_attr\_set\_ftp (globus\_gass\_copy\_attr\_t \* attr, globus\_ftp\_client\_operationattr\_t \* ftp\_attr)**

Set the attributes for ftp/gsiftp transfers.

In order to specify attributes for ftp/gsiftp transfers, a globus\_ftp\_client\_operationattr\_t should be initialized and its values set using the appropriate globus\_ftp\_client\_operationattr\_\* functions. The globus\_ftp\_client\_operationattr\_t \* can then be passed to the globus\_gass\_copy\_attr\_t via this function.

##### **Parameters:**

*attr* A globus\_gass\_copy attribute structure

*ftp\_attr* The ftp/gsiftp attributes to be used

##### **Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

##### **See also:**

[globus\\_gass\\_copy\\_attr\\_init\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_gass\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_io\(\)](#), [globus\\_gass\\_copy\\_get\\_url\\_mode\(\)](#), [globus\\_ftp\\_client\\_operationattr\\_\\*](#)

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.12 globus\_result\_t globus\_gass\_copy\_attr\_set\_io (globus\_gass\_copy\_attr\_t \* attr, globus\_io\_attr\_t \* io\_attr)**

Set the attributes for file transfers.

In order to specify attributes for file transfers, a globus\_io\_attr\_t should be initialized and its values set using the appropriate globus\_io\_attr\_\* functions. The globus\_io\_attr\_t can then be passed to the globus\_gass\_copy\_attr\_t via this function.

##### **Parameters:**

*attr* A globus\_gass\_copy attribute structure

*io\_attr* The file attributes to be used

##### **Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

##### **See also:**

[globus\\_gass\\_copy\\_attr\\_init\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_gass\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_ftp\(\)](#), [globus\\_gass\\_copy\\_get\\_url\\_mode\(\)](#), [globus\\_io\\_attr\\_\\*](#)

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.13 globus\_result\_t globus\_gass\_copy\_attr\_set\_gass (globus\_gass\_copy\_attr\_t \* attr, globus\_gass\_transfer\_requestattr\_t \* gass\_attr)**

Set the attributes for http/https transfers.

In order to specify attributes for http/https transfers, a globus\_gass\_transfer\_requestattr\_t should be initialized and its values set using the appropriate globus\_gass\_transfer\_requestattr\_\* functions. The globus\_gass\_transfer\_requestattr\_t can then be passed to the globus\_gass\_copy\_attr\_t via this function.

**Parameters:**

*attr* A globus\_gass\_copy attribute structure  
*gass\_attr* The http/https attributes to be used

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

**See also:**

[globus\\_gass\\_copy\\_attr\\_init\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_io\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_ftp\(\)](#), [globus\\_gass\\_copy\\_get\\_url\\_mode\(\)](#), [globus\\_gass\\_transfer\\_requestattr\\_\\*](#)

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.14 globus\_result\_t globus\_gass\_copy\_get\_url\_mode (char \* url, globus\_gass\_copy\_url\_mode\_t \* mode)**

Classify the URL schema into the transfer method that will be used to do the actual transfer.

This function enables the user to determine what protocol will be used to transfer data to/from a particular url. This information can then be used to specify the appropriate attributes when initiating a transfer.

**Parameters:**

*url* The URL for schema checking  
*mode* the filled in schema type of the URL param

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

**See also:**

[globus\\_gass\\_copy\\_attr\\_init\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_io\(\)](#), [globus\\_gass\\_copy\\_attr\\_set\\_ftp\(\)](#), [globus\\_gass\\_copy\\_set\\_gass\(\)](#)

References GLOBUS\_GASS\_COPY\_MODULE.

**5.1.2.15 globus\_result\_t globus\_gass\_copy\_register\_performance\_cb (globus\_gass\_copy\_handle\_t \* handle, globus\_gass\_copy\_performance\_cb\_t callback, void \* user\_arg)**

Register a performance information callback.

Use this to register a performance information callback. You change or set to GLOBUS\_NULL the callback any time a transfer is not occurring.

**Parameters:**

*handle* an initialized gass copy handle for which you would like to see performance info  
*callback* the performance callback  
*user\_arg* a user pointer that will be passed to all callbacks for a given handle

**Returns:**

- GLOBUS\_SUCCESS

- error on a NULL or busy handle

See also:

[globus\\_gass\\_copy\\_performance\\_cb\\_t](#)

References GLOBUS\_GASS\_COPY\_MODULE.

#### 5.1.2.16 **globus\_result\_t globus\_gass\_copy\_get\_status (globus\_gass\_copy\_handle\_t \* *handle*, globus\_gass\_copy\_status\_t \* *status*)**

Get the status code of the current transfer.

Get the status of the last transfer to be initiated using the given handle. Only one transfer can be active on a handle at a given time, therefore new transfers may only be initiated when the current status is one of the following: GLOBUS\_GASS\_COPY\_STATUS\_NONE, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_SUCCESS, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_FAILURE, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_CANCELLED

**Parameters:**

***handle*** A globus\_gass\_copy\_handle

***status*** Will be one of the following: GLOBUS\_GASS\_COPY\_STATUS\_NONE (No transfers have been initiated using this handle.) GLOBUS\_GASS\_COPY\_STATUS\_PENDING (A transfer is currently being set up.) GLOBUS\_GASS\_COPY\_STATUS\_TRANSFER\_IN\_PROGRESS (There is currently a transfer in progress.) GLOBUS\_GASS\_COPY\_STATUS\_CANCEL (The last transfer initiated using this handle has been cancelled by the user before completing, and is in the process of being cleaned up.) GLOBUS\_GASS\_COPY\_STATUS\_FAILURE (The last transfer initiated using this handle failed, and is in the process of being cleaned up.) GLOBUS\_GASS\_COPY\_STATUS\_DONE\_SUCCESS (The last transfer initiated using this handle has completed successfully.) GLOBUS\_GASS\_COPY\_STATUS\_DONE\_FAILURE (The last transfer initiated using this handle failed and has finished cleaning up.) GLOBUS\_GASS\_COPY\_STATUS\_DONE\_CANCELLED (The last transfer initiated using this handle was cancelled and has finished cleaning up.)

**Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

#### 5.1.2.17 **const char\* globus\_gass\_copy\_get\_status\_string (globus\_gass\_copy\_handle\_t \* *handle*)**

Get the status string of the current transfer.

Get the status of the last transfer to be initiated using the given handle. Only one transfer can be active on a handle at a given time, therefore new transfers may only be initiated when the current status is one of the following: GLOBUS\_GASS\_COPY\_STATUS\_NONE, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_SUCCESS, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_FAILURE, GLOBUS\_GASS\_COPY\_STATUS\_DONE\_CANCELLED

**Parameters:**

***handle*** A globus\_gass\_copy\_handle

**Returns:**

Returns a pointer to a character string describing the current status

References globus\_gass\_copy\_get\_status().

**5.1.2.18** `globus_result_t globus_gass_copy_url_to_url (globus_gass_copy_handle_t * handle, char * source_url, globus_gass_copy_attr_t * source_attr, char * dest_url, globus_gass_copy_attr_t * dest_attr)`

Transfer data from source URL to destination URL (blocking).

**Parameters:**

*handle* The handle to perform the copy operation

*source\_url* transfer data from this URL

*source\_attr* Attributes describing how the transfer from the source should be done

*dest\_url* transfer data to this URL

*dest\_attr* Attributes describing how the transfer to the destination should be done

**Returns:**

This function returns GLOBUS\_SUCCESS if the transfer was completed successfully, or a result pointing to an object of one of the the following error types:

**Return values:**

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_NULL\_PARAMETER** The handle was equal to GLOBUS\_NULL, so the transfer could not processed.

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_next\_error** next error description

**See also:**

[globus\\_gass\\_copy\\_url\\_to\\_handle\(\)](#) [globus\\_gass\\_copy\\_handle\\_to\\_url\(\)](#)

References `globus_i_gass_copy_monitor_t::cond`, `globus_i_gass_copy_monitor_t::done`, `globus_i_gass_copy_monitor_t::err`, `GLOBUS_GASS_COPY_MODULE`, `globus_gass_copy_register_url_to_url()`, `globus_i_gass_copy_monitor_t::mutex`, and `globus_i_gass_copy_monitor_t::use_err`.

**5.1.2.19** `globus_result_t globus_gass_copy_url_to_handle (globus_gass_copy_handle_t * handle, char * source_url, globus_gass_copy_attr_t * source_attr, globus_io_handle_t * dest_handle)`

Transfer data from source URL to an IO handle (blocking).

**Parameters:**

*handle* The handle to perform the copy operation

*source\_url* transfer data from this URL

*source\_attr* Attributes describing how the transfer from the source should be done

*dest\_handle* transfer data to this IO handle

**Returns:**

This function returns GLOBUS\_SUCCESS if the transfer was completed successfully, or a result pointing to an object of one of the the following error types:

**Return values:**

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_NULL\_PARAMETER** The handle was equal to GLOBUS\_NULL, so the transfer could not processed.

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_next\_error** next error description



See also:

[globus\\_gass\\_copy\\_url\\_to\\_url\(\)](#) [globus\\_gass\\_copy\\_handle\\_to\\_url\(\)](#)

References `globus_i_gass_copy_monitor_t::cond`, `globus_i_gass_copy_monitor_t::done`, `globus_i_gass_copy_monitor_t::err`, `GLOBUS_GASS_COPY_MODULE`, `globus_gass_copy_register_url_to_handle()`, `globus_i_gass_copy_monitor_t::mutex`, and `globus_i_gass_copy_monitor_t::use_err`.

**5.1.2.20 `globus_result_t globus_gass_copy_handle_to_url (globus_gass_copy_handle_t * handle, globus_io_handle_t * source_handle, char * dest_url, globus_gass_copy_attr_t * dest_attr)`**

Transfer data from an IO handle to destination URL (blocking).

**Parameters:**

*handle* The handle to perform the copy operation

*source\_handle* transfer data from this IO handle

*dest\_url* transfer data to this URL

*dest\_attr* Attributes describing how the transfer to the destination should be done

**Returns:**

This function returns `GLOBUS_SUCCESS` if the transfer was completed successfully, or a result pointing to an object of one of the the following error types:

**Return values:**

**`GLOBUS_GASS_COPY_ERROR_TYPE_NULL_PARAMETER`** The handle was equal to `GLOBUS_NULL`, so the transfer could not processed.

**`GLOBUS_GASS_COPY_ERROR_TYPE_next_error`** next error description

See also:

[globus\\_gass\\_copy\\_url\\_to\\_url\(\)](#) [globus\\_gass\\_copy\\_url\\_to\\_handle\(\)](#)

References `globus_i_gass_copy_monitor_t::cond`, `globus_i_gass_copy_monitor_t::done`, `globus_i_gass_copy_monitor_t::err`, `GLOBUS_GASS_COPY_MODULE`, `globus_gass_copy_register_handle_to_url()`, `globus_i_gass_copy_monitor_t::mutex`, and `globus_i_gass_copy_monitor_t::use_err`.

**5.1.2.21 `globus_result_t globus_gass_copy_register_url_to_url (globus_gass_copy_handle_t * handle, char * source_url, globus_gass_copy_attr_t * source_attr, char * dest_url, globus_gass_copy_attr_t * dest_attr, globus_gass_copy_callback_t callback_func, void * callback_arg)`**

Transfer data from source URL to destination URL (non-blocking).

This functions initiates a transfer from source URL to destination URL, then returns immediately.

When the transfer is completed or if the transfer is aborted, the `callback_func` will be invoked with the final status of the transfer.

**Parameters:**

*handle* The handle to perform the copy operation

*source\_url* transfer data from this URL

*source\_attr* Attributes describing how the transfer form the source should be done

*dest\_url* transfer data to this URL

*dest\_attr* Attributes describing how the transfer to the destination should be done

*callback\_func* Callback to be invoked once the transfer is completed.

*callback\_arg* Argument to be passed to the callback\_func.

#### Returns:

This function returns GLOBUS\_SUCCESS if the transfer was initiated successfully, or a result pointing to an object of one of the the following error types:

#### Return values:

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_NULL\_PARAMETER** The handle was equal to GLOBUS\_NULL, so the transfer could not processed.

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_next\_error** next error description

#### See also:

[globus\\_gass\\_copy\\_register\\_url\\_to\\_handle\(\)](#), [globus\\_gass\\_copy\\_register\\_handle\\_to\\_url\(\)](#)

References `globus_i_gass_copy_state_target_s::attr`, `globus_gass_copy_state_s::cancel`, `globus_gass_copy_state_s::dest`, `globus_gass_copy_get_url_mode()`, `GLOBUS_GASS_COPY_MODULE`, and `globus_gass_copy_state_s::source`.

**5.1.2.22** `globus_result_t globus_gass_copy_register_url_to_handle (globus_gass_copy_handle_t * handle, char * source_url, globus_gass_copy_attr_t * source_attr, globus_io_handle_t * dest_handle, globus_gass_copy_callback_t callback_func, void * callback_arg)`

Transfer data from source URL to an IO handle (non-blocking).

This functions initiates a transfer from source URL to an IO handle, then returns immediately.

When the transfer is completed or if the transfer is aborted, the callback\_func will be invoked with the final status of the transfer.

#### Parameters:

*handle* The handle to perform the copy operation

*source\_url* transfer data from this URL

*source\_attr* Attributes describing how the transfer form the source should be done

*dest\_handle* transfer data to this IO handle

*callback\_func* Callback to be invoked once the transfer is completed.

*callback\_arg* Argument to be passed to the callback\_func.

#### Returns:

This function returns GLOBUS\_SUCCESS if the transfer was initiated successfully, or a result pointing to an object of one of the the following error types:

#### Return values:

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_NULL\_PARAMETER** The handle was equal to GLOBUS\_NULL, so the transfer could not processed.

**GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_next\_error** next error description

#### See also:

[globus\\_gass\\_copy\\_register\\_url\\_to\\_url\(\)](#), [globus\\_gass\\_copy\\_register\\_handle\\_to\\_url\(\)](#)

References `globus_gass_copy_state_s::cancel`, `globus_gass_copy_state_s::dest`, `globus_gass_copy_get_url_mode()`, `GLOBUS_GASS_COPY_MODULE`, and `globus_gass_copy_state_s::source`.

**5.1.2.23** `globus_result_t globus_gass_copy_register_handle_to_url (globus_gass_copy_handle_t * handle, globus_io_handle_t * source_handle, char * dest_url, globus_gass_copy_attr_t * dest_attr, globus_gass_copy_callback_t callback_func, void * callback_arg)`

Transfer data from an IO handle to destination URL (non-blocking).

This functions initiates a transfer from an IO handle to destination URL, then returns immediately.

When the transfer is completed or if the transfer is aborted, the `callback_func` will be invoked with the final status of the transfer.

**Parameters:**

*handle* The handle to perform the copy operation

*source\_handle* transfer data from this IO handle

*dest\_url* transfer data to this URL

*dest\_attr* Attributes describing how the transfer to the destination should be done

*callback\_func* Callback to be invoked once the transfer is completed.

*callback\_arg* Argument to be passed to the `callback_func`.

**Returns:**

This function returns `GLOBUS_SUCCESS` if the transfer was initiated successfully, or a result pointing to an object of one of the the following error types:

**Return values:**

***GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_NULL\_PARAMETER*** The handle was equal to `GLOBUS_NULL`, so the transfer could not processed.

***GLOBUS\_GASS\_COPY\_ERROR\_TYPE\_next\_error*** next error description

**See also:**

[globus\\_gass\\_copy\\_register\\_url\\_to\\_url\(\)](#), [globus\\_gass\\_copy\\_register\\_url\\_to\\_handle\(\)](#)

References `globus_gass_copy_state_s::cancel`, `globus_gass_copy_state_s::dest`, `globus_gass_copy_get_url_mode()`, `GLOBUS_GASS_COPY_MODULE`, and `globus_gass_copy_state_s::source`.

**5.1.2.24** `globus_result_t globus_gass_copy_cache_url_state (globus_gass_copy_handle_t * handle, char * url)`

Cache connections to an FTP or GSIFTP server.

Explicitly cache connections to URL server. When an URL is cached, the connection to the URL server will not be closed after a file transfer completes.

**Parameters:**

*handle* Handle which will contain a cached connection to the URL server.

*url* The URL of the FTP or GSIFTP server to cache.

**Returns:**

This function returns `GLOBUS_SUCCESS` if successful, or a `globus_result_t` indicating the error that occurred.

References `GLOBUS_GASS_COPY_MODULE`.

#### **5.1.2.25 globus\_result\_t globus\_gass\_copy\_flush\_url\_state (globus\_gass\_copy\_handle\_t \* *handle*, char \* *url*)**

Remove a cached connection to an FTP or GSIFTP server.

Explicitly remove a cached connection to an FTP or GSIFTP server. If an idle connection to an FTP server exists, it will be closed.

##### **Parameters:**

*handle* Handle which contains a cached connection to the URL server.

*url* The URL of the FTP or GSIFTP server to remove.

##### **Returns:**

This function returns GLOBUS\_SUCCESS if successful, or a globus\_result\_t indicating the error that occurred.

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.26 globus\_result\_t globus\_gass\_copy\_set\_user\_pointer (globus\_gass\_copy\_handle\_t \* *handle*, void \* *user\_pointer*)**

Set a pointer in the handle to point at user-allocated memory.

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.27 globus\_result\_t globus\_gass\_copy\_get\_user\_pointer (globus\_gass\_copy\_handle\_t \* *handle*, void \*\* *user\_data*)**

Get the pointer in the handle that points to user-allocated memory.

References GLOBUS\_GASS\_COPY\_MODULE.

#### **5.1.2.28 globus\_result\_t globus\_gass\_copy\_cancel (globus\_gass\_copy\_handle\_t \* *handle*, globus\_gass\_copy\_callback\_t *cancel\_callback*, void \* *cancel\_callback\_arg*)**

Cancel the current transfer associated with this handle.

store the cancel\_callback and cancel\_callback\_arg in the handle. Needed because the ftp callback will be the one given from the original globus\_ftp\_client\_third\_party\_transfer() call.

References globus\_i\_gass\_copy\_cancel\_s::canceling\_source, GLOBUS\_GASS\_COPY\_MODULE, globus\_l\_gass\_copy\_target\_cancel(), and globus\_i\_gass\_copy\_cancel\_s::handle.

#### **5.1.2.29 globus\_result\_t globus\_l\_gass\_copy\_target\_cancel (globus\_i\_gass\_copy\_cancel\_t \* *cancel\_info*)**

Cancel the source or destination transfer in progress.

References globus\_i\_gass\_copy\_cancel\_s::canceling\_source, globus\_i\_gass\_copy\_state\_target\_s::data, globus\_i\_gass\_copy\_state\_target\_s::ftp, globus\_i\_gass\_copy\_state\_target\_s::gass, GLOBUS\_GASS\_COPY\_MODULE, globus\_i\_gass\_copy\_cancel\_s::handle, globus\_i\_gass\_copy\_state\_target\_s::io, globus\_i\_gass\_copy\_state\_target\_s::mode, and globus\_i\_gass\_copy\_state\_target\_s::url.

## **5.2 globus\_gass\_copy.h File Reference**

Header file for the gass copy library.

## Data Structures

- struct `globus_gass_copy_glob_stat_t`  
*Glob expanded entry information.*

## Defines

- #define `GLOBUS_GASS_COPY_MODULE` (`&globus_i_gass_copy_module`)

## Typedefs

- typedef void(\* `globus_gass_copy_performance_cb_t` )(void \*user\_arg, globus\_gass\_copy\_handle\_t \*handle, globus\_off\_t total\_bytes, float instantaneous\_throughput, float avg\_throughput)
- typedef void(\* `globus_gass_copy_glob_entry_cb_t` )(const char \*url, const `globus_gass_copy_glob_stat_t` \*info\_stat, void \*user\_arg)

## Enumerations

- enum `globus_gass_copy_glob_entry_t`

## Functions

- globus\_result\_t `globus_gass_copy_glob_expand_url` (globus\_gass\_copy\_handle\_t \*handle, const char \*url, globus\_gass\_copy\_attr\_t \*attr, `globus_gass_copy_glob_entry_cb_t` entry\_cb, void \*user\_arg)
- globus\_result\_t `globus_gass_copy_mkdir` (globus\_gass\_copy\_handle\_t \*handle, char \*url, globus\_gass\_copy\_attr\_t \*attr)

### 5.2.1 Detailed Description

Header file for the gass copy library.

### 5.2.2 Define Documentation

#### 5.2.2.1 #define GLOBUS\_GASS\_COPY\_MODULE (&globus\_i\_gass\_copy\_module)

Module descriptor.

Globus GASS Copy uses standard Globus module activation and deactivation. Before any Globus GASS Copy functions are called, the following function must be called:

```
globus_module_activate(GLOBUS_GASS_COPY_MODULE)
```

This function returns `GLOBUS_SUCCESS` if Globus GASS Copy was successfully initialized, and you are therefore allowed to subsequently call Globus GASS Copy functions. Otherwise, an error code is returned, and Globus GASS Copy functions should not be subsequently called. This function may be called multiple times.

To deactivate Globus GASS Copy, the following function must be called:

```
globus_module_deactivate(GLOBUS_GASS_COPY_MODULE)
```

This function should be called once for each time Globus GASS Copy was activated.

### 5.2.3 Typedef Documentation

**5.2.3.1** `typedef void(* globus_gass_copy_performance_cb_t)(void *user_arg, globus_gass_copy_handle_t *handle, globus_off_t total_bytes, float instantaneous_throughput, float avg_throughput)`

Gass copy transfer performance callback.

This callback is registered with 'globus\_gass\_copy\_register\_performance\_cb'. It will be called during a transfer to supply performance information on current transfer. Its frequency will be at most one per second, but it is possible to receive no callbacks. This is possible in very short transfers and in ftp transfers in which the server does not provide performance information.

#### Parameters:

- handle* the gass copy handle this transfer is occurring on
- user\_arg* a user pointer registered with 'globus\_gass\_copy\_register\_performance\_cb'
- total\_bytes* the total number of bytes transfer so far
- instantaneous\_throughput* instantaneous rate of transfer (since last callback or start) (bytes / sec)
- avg\_throughput* the avg throughput calculated since the start of the transfer (bytes / sec)

#### Returns:

- n/a

**5.2.3.2** `typedef void(* globus_gass_copy_glob_entry_cb_t)(const char *url, const globus_gass_copy_glob_stat_t *info_stat, void *user_arg)`

Gass copy glob entry callback.

This callback is passed as a parameter to [globus\\_gass\\_copy\\_glob\\_expand\\_url\(\)](#). It is called once for each entry that the original expands to.

#### Parameters:

- url* The full url to the expanded entry. A directory entry will end in a forward slash '/'.
- stat* A pointer to a [globus\\_gass\\_copy\\_glob\\_stat\\_t](#) containing information about the entry.
- user\_arg* The user\_arg passed to [globus\\_gass\\_copy\\_glob\\_expand\(\)](#)

#### See also:

[globus\\_gass\\_copy\\_glob\\_stat\\_t](#), [globus\\_gass\\_copy\\_glob\\_expand\\_url](#)

### 5.2.4 Enumeration Type Documentation

**5.2.4.1** `enum globus_gass_copy_glob_entry_t`

globbed entry types

### 5.2.5 Function Documentation

**5.2.5.1** `globus_result_t globus_gass_copy_glob_expand_url(globus_gass_copy_handle_t *handle, const char *url, globus_gass_copy_attr_t *attr, globus_gass_copy_glob_entry_cb_t entry_cb, void *user_arg)`

Expand globbed url.

This function expands wildcards in a globbed url, and calls entry\_cb() on each one.

**Parameters:**

**handle** A gass copy handle to use for the operation.

**url** The URL to expand. The URL may be an ftp, gsiftp or file URL. Wildcard characters supported are '\*' '[' ']' in the filename portion of the url.

**attr** Gass copy attributes for this operation.

**entry\_cb** Function to call with information about each entry

**user\_arg** An argument to pass to entry\_cb()

**Returns:**

This function returns an error when any of these conditions are true:

- handle is GLOBUS\_NULL
- url is GLOBUS\_NULL
- url cannot be parsed
- url is not a ftp, gsiftp or file url

References globus\_gass\_copy\_glob\_expand\_url(), GLOBUS\_GASS\_COPY\_MODULE, globus\_gass\_copy\_glob\_stat\_t::mdtm, globus\_gass\_copy\_glob\_stat\_t::mode, globus\_gass\_copy\_glob\_stat\_t::size, globus\_gass\_copy\_glob\_stat\_t::symlink\_target, globus\_gass\_copy\_glob\_stat\_t::type, and globus\_gass\_copy\_glob\_stat\_t::unique\_id.

### 5.2.5.2 globus\_result\_t globus\_gass\_copy\_mkdir (globus\_gass\_copy\_handle\_t \* handle, char \* url, globus\_gass\_copy\_attr\_t \* attr)

Make directory.

This function creates a directory given a ftp or file url.

**Parameters:**

**handle** A gass copy handle to use for the mkdir operation.

**url** The URL for the directory to create. The URL may be an ftp, gsiftp or file URL.

**attr** Gass copy attributes for this operation.

**Returns:**

This function returns an error when any of these conditions are true:

- handle is GLOBUS\_NULL
- url is GLOBUS\_NULL
- url cannot be parsed
- url is not a ftp, gsiftp or file url
- the directory could not be created

References globus\_gass\_copy\_get\_url\_mode(), globus\_gass\_copy\_mkdir(), and GLOBUS\_GASS\_COPY\_MODULE.

## Index

- attr
  - globus\_i\_gass\_copy\_state\_target\_s, 4
- cancel
  - globus\_gass\_copy\_state\_s, 3
- data
  - globus\_i\_gass\_copy\_state\_target\_s, 5
- dest
  - globus\_gass\_copy\_state\_s, 3
- free\_handle
  - globus\_i\_gass\_copy\_state\_target\_s, 5
- ftp
  - globus\_i\_gass\_copy\_state\_target\_s, 5
- gass
  - globus\_i\_gass\_copy\_state\_target\_s, 5
- globus\_gass\_copy.c, 6
  - globus\_gass\_copy\_attr\_init, 10
  - globus\_gass\_copy\_attr\_set\_ftp, 10
  - globus\_gass\_copy\_attr\_set\_gass, 11
  - globus\_gass\_copy\_attr\_set\_io, 11
  - globus\_gass\_copy\_cache\_url\_state, 17
  - globus\_gass\_copy\_cancel, 18
  - globus\_gass\_copy\_flush\_url\_state, 17
  - globus\_gass\_copy\_get\_buffer\_length, 8
  - globus\_gass\_copy\_get\_no\_third\_party\_transfers, 9
  - globus\_gass\_copy\_get\_partial\_offsets, 10
  - globus\_gass\_copy\_get\_status, 13
  - globus\_gass\_copy\_get\_status\_string, 13
  - globus\_gass\_copy\_get\_url\_mode, 12
  - globus\_gass\_copy\_get\_user\_pointer, 18
  - globus\_gass\_copy\_handle\_destroy, 7
  - globus\_gass\_copy\_handle\_init, 7
  - globus\_gass\_copy\_handle\_to\_url, 15
  - globus\_gass\_copy\_register\_handle\_to\_url, 16
  - globus\_gass\_copy\_register\_performance\_cb, 12
  - globus\_gass\_copy\_register\_url\_to\_handle, 16
  - globus\_gass\_copy\_register\_url\_to\_url, 15
  - globus\_gass\_copy\_set\_allocate, 9
  - globus\_gass\_copy\_set\_buffer\_length, 8
  - globus\_gass\_copy\_set\_no\_third\_party\_transfers, 8
  - globus\_gass\_copy\_set\_partial\_offsets, 9
  - globus\_gass\_copy\_set\_user\_pointer, 18
  - globus\_gass\_copy\_url\_to\_handle, 14
  - globus\_gass\_copy\_url\_to\_url, 13
  - globus\_l\_gass\_copy\_target\_cancel, 18
- globus\_gass\_copy.h, 18
  - globus\_gass\_copy\_glob\_entry\_cb\_t, 20
  - globus\_gass\_copy\_glob\_entry\_t, 20
  - globus\_gass\_copy\_glob\_expand\_url, 20
  - globus\_gass\_copy\_mkdir, 21
  - GLOBUS\_GASS\_COPY\_MODULE, 19
  - globus\_gass\_copy\_performance\_cb\_t, 20
  - globus\_gass\_copy\_attr\_init
    - globus\_gass\_copy.c, 10
  - globus\_gass\_copy\_attr\_set\_ftp
    - globus\_gass\_copy.c, 10
  - globus\_gass\_copy\_attr\_set\_gass
    - globus\_gass\_copy.c, 11
  - globus\_gass\_copy\_attr\_set\_io
    - globus\_gass\_copy.c, 11
  - globus\_gass\_copy\_cache\_url\_state
    - globus\_gass\_copy.c, 17
  - globus\_gass\_copy\_cancel
    - globus\_gass\_copy.c, 18
  - globus\_gass\_copy\_flush\_url\_state
    - globus\_gass\_copy.c, 17
  - globus\_gass\_copy\_get\_buffer\_length
    - globus\_gass\_copy.c, 8
  - globus\_gass\_copy\_get\_no\_third\_party\_transfers
    - globus\_gass\_copy.c, 9
  - globus\_gass\_copy\_get\_partial\_offsets
    - globus\_gass\_copy.c, 10
  - globus\_gass\_copy\_get\_status
    - globus\_gass\_copy.c, 13
  - globus\_gass\_copy\_get\_status\_string
    - globus\_gass\_copy.c, 13
  - globus\_gass\_copy\_get\_url\_mode
    - globus\_gass\_copy.c, 12
  - globus\_gass\_copy\_get\_user\_pointer
    - globus\_gass\_copy.c, 18
  - globus\_gass\_copy\_glob\_entry\_cb\_t
    - globus\_gass\_copy.h, 20
  - globus\_gass\_copy\_glob\_entry\_t
    - globus\_gass\_copy.h, 20
  - globus\_gass\_copy\_glob\_expand\_url
    - globus\_gass\_copy.h, 20
  - globus\_gass\_copy\_glob\_stat\_t, 1
    - mdtm, 2
    - mode, 2
    - size, 2
    - symlink\_target, 2
    - type, 2
    - unique\_id, 2
  - globus\_gass\_copy\_handle\_destroy
    - globus\_gass\_copy.c, 7
  - globus\_gass\_copy\_handle\_init
    - globus\_gass\_copy.c, 7
  - globus\_gass\_copy\_handle\_to\_url
    - globus\_gass\_copy.c, 15
  - globus\_gass\_copy\_mkdir
    - globus\_gass\_copy.h, 21



GLOBUS\_GASS\_COPY\_MODULE  
     globus\_gass\_copy.h, [19](#)  
 globus\_gass\_copy\_performance\_cb\_t  
     globus\_gass\_copy.h, [20](#)  
 globus\_gass\_copy\_register\_handle\_to\_url  
     globus\_gass\_copy.c, [16](#)  
 globus\_gass\_copy\_register\_performance\_cb  
     globus\_gass\_copy.c, [12](#)  
 globus\_gass\_copy\_register\_url\_to\_handle  
     globus\_gass\_copy.c, [16](#)  
 globus\_gass\_copy\_register\_url\_to\_url  
     globus\_gass\_copy.c, [15](#)  
 globus\_gass\_copy\_set\_allocate  
     globus\_gass\_copy.c, [9](#)  
 globus\_gass\_copy\_set\_buffer\_length  
     globus\_gass\_copy.c, [8](#)  
 globus\_gass\_copy\_set\_no\_third\_party\_transfers  
     globus\_gass\_copy.c, [8](#)  
 globus\_gass\_copy\_set\_partial\_offsets  
     globus\_gass\_copy.c, [9](#)  
 globus\_gass\_copy\_set\_user\_pointer  
     globus\_gass\_copy.c, [18](#)  
 globus\_gass\_copy\_state\_s, [2](#)  
     cancel, [3](#)  
     dest, [3](#)  
     monitor, [3](#)  
     mutex, [3](#)  
     source, [3](#)  
 globus\_gass\_copy\_url\_to\_handle  
     globus\_gass\_copy.c, [14](#)  
 globus\_gass\_copy\_url\_to\_url  
     globus\_gass\_copy.c, [13](#)  
 globus\_i\_gass\_copy\_buffer\_t, [3](#)  
 globus\_i\_gass\_copy\_cancel\_s, [3](#)  
 globus\_i\_gass\_copy\_monitor\_t, [4](#)  
 globus\_i\_gass\_copy\_state\_target\_s, [4](#)  
     attr, [4](#)  
     data, [5](#)  
     free\_handle, [5](#)  
     ftp, [5](#)  
     gass, [5](#)  
     io, [5](#)  
     mode, [5](#)  
     mutex, [4](#)  
     n\_complete, [5](#)  
     n\_pending, [5](#)  
     n\_simultaneous, [5](#)  
     queue, [5](#)  
     request, [5](#)  
     seekable, [5](#)  
     status, [5](#)  
     url, [4](#)  
 globus\_l\_gass\_copy\_target\_cancel  
     globus\_gass\_copy.c, [18](#)  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 mdtm  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
 mode  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 monitor  
     globus\_gass\_copy\_state\_s, [3](#)  
 mutex  
     globus\_gass\_copy\_state\_s, [3](#)  
     globus\_i\_gass\_copy\_state\_target\_s, [4](#)  
 n\_complete  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 n\_pending  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 n\_simultaneous  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 queue  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 request  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 seekable  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 size  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
 source  
     globus\_gass\_copy\_state\_s, [3](#)  
 status  
     globus\_i\_gass\_copy\_state\_target\_s, [5](#)  
 symlink\_target  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
 type  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
 unique\_id  
     globus\_gass\_copy\_glob\_stat\_t, [2](#)  
 url  
     globus\_i\_gass\_copy\_state\_target\_s, [4](#)

io