

libcarddav-0.6.1

Generated by Doxygen 1.9.5



<b>1 libcarddav-0.6.1</b>	<b>1</b>
<b>2 Deprecated List</b>	<b>3</b>
<b>3 Data Structure Index</b>	<b>5</b>
3.1 Data Structures . . . . .	5
<b>4 File Index</b>	<b>7</b>
4.1 File List . . . . .	7
<b>5 Data Structure Documentation</b>	<b>9</b>
5.1 _carddav_error Struct Reference . . . . .	9
5.1.1 Detailed Description . . . . .	9
5.1.2 Field Documentation . . . . .	9
5.1.2.1 str . . . . .	9
5.2 _CARDDAV_SETTINGS Struct Reference . . . . .	10
5.2.1 Detailed Description . . . . .	10
5.3 _response Struct Reference . . . . .	10
5.3.1 Detailed Description . . . . .	10
5.3.2 Field Documentation . . . . .	11
5.3.2.1 msg . . . . .	11
5.4 config_data Struct Reference . . . . .	11
5.4.1 Detailed Description . . . . .	11
5.5 debug_curl Struct Reference . . . . .	11
5.5.1 Field Documentation . . . . .	12
5.5.1.1 debug . . . . .	12
5.5.1.2 trace_ascii . . . . .	12
5.6 MD5_CONTEXT Struct Reference . . . . .	12
5.7 MemoryStruct Struct Reference . . . . .	12
5.7.1 Detailed Description . . . . .	13
5.8 runtime_info Struct Reference . . . . .	13
<b>6 File Documentation</b>	<b>15</b>
6.1 carddav.h File Reference . . . . .	15
6.1.1 Detailed Description . . . . .	17
6.1.2 Enumeration Type Documentation . . . . .	17
6.1.2.1 CARDDAV_ACTION . . . . .	17
6.1.2.2 CARDDAV_RESPONSE . . . . .	17
6.1.3 Function Documentation . . . . .	17
6.1.3.1 carddav_add_object() . . . . .	17
6.1.3.2 carddav_delete_object() . . . . .	18
6.1.3.3 carddav_delete_object_by_uri() . . . . .	19
6.1.3.4 carddav_enabled_resource() . . . . .	20
6.1.3.5 carddav_free_error() . . . . .	21

6.1.3.6 carddav_free_response()	21
6.1.3.7 carddav_free_runtime_info()	22
6.1.3.8 carddav_get_displayname()	22
6.1.3.9 carddav_get_error()	23
6.1.3.10 carddav_get_freebusy()	24
6.1.3.11 carddav_get_object()	25
6.1.3.12 carddav_get_response()	26
6.1.3.13 carddav_get_runtime_info()	26
6.1.3.14 carddav_get_server_options()	26
6.1.3.15 carddav_getall_object()	27
6.1.3.16 carddav_getall_object_by_uri()	28
6.1.3.17 carddav_modify_object()	30
6.1.3.18 carddav_modify_object_by_uri()	31
6.1.3.19 carddav_set_options()	32

# Chapter 1

## libcarddav-0.6.1

This document is the documentation for the public interface to libcarddav. If you want to study the implementation look for the developers API.

The libray and documentation is Copyright (c) 2010 Timothy Pearson ( [kb9vqf@pearsoncomputing.net](mailto:kb9vqf@pearsoncomputing.net)) and the original caldav implementation is copyright (c) 2008 Michael Rasmussen ( [mir@datanom.net](mailto:mir@datanom.net))

License for the source code.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

License for the documentation.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.



## Chapter 2

# Deprecated List

### Global `carddav_get_error` (`carddav_error *lib_error`)

Function to call in case of errors. Caller provides a pointer to a local `carddav_error` structure. `Carddav_get_error` will initialize pointer if NULL. Caller is responsible for freeing returned memory. After the first call the internal error buffer is reset.

Always returns an initialized empty `carddav_error` Function to call in case of errors. Caller provides a pointer to a local `carddav_error` structure. `Carddav_get_error` will initialize pointer if NULL. Caller is responsible for freeing returned memory. After the first call the internal error buffer is reset.

### Global `carddav_set_options` (`debug_curl curl_options`)

Does nothing Function which supports sending various options inside the library.





## Chapter 3

# Data Structure Index

### 3.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">_carddav_error</a>	A struct for storing error codes and messages . . . . .	9
<a href="#">_CARDDAV_SETTINGS</a>	A struct used to exchange all user input between various parts of the library . . . . .	10
<a href="#">_response</a>	A struct used for returning messages from the library to users . . . . .	10
<a href="#">config_data</a>	Used to exchange user options to the library . . . . .	11
<a href="#">debug_curl</a>	. . . . .	11
<a href="#">MD5_CONTEXT</a>	. . . . .	12
<a href="#">MemoryStruct</a>	Used to hold messages between the CardDAV server and the library . . . . .	12
<a href="#">runtime_info</a>	. . . . .	13



## Chapter 4

# File Index

### 4.1 File List

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

<a href="#">carddav.h</a>	Interface to the carddav library . . . . .	15
---------------------------	--	----



## Chapter 5

# Data Structure Documentation

### 5.1 `_carddav_error` Struct Reference

A struct for storing error codes and messages.

```
#include <carddav.h>
```

#### Data Fields

- long **code**  
*if < 0 internal error > 0 CardDAV protocol error.*
- char \* [str](#)  
*For storing human readable error message.*

#### 5.1.1 Detailed Description

A struct for storing error codes and messages.

#### 5.1.2 Field Documentation

##### 5.1.2.1 `str`

```
char * _carddav_error::str
```

For storing human readable error message.

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

- [carddav.h](#)

## 5.2 `_CARDDAV_SETTINGS` Struct Reference

A struct used to exchange all user input between various parts of the library.

```
#include <carddav-utils.h>
```

### Data Fields

- `gchar * username`
- `gchar * password`
- `gchar * url`
- `gchar * file`
- `gboolean usehttps`
- `gboolean verify_ssl_certificate`
- `gchar * custom_cacert`
- `gboolean debug`
- `gboolean use_locking`
- `char trace_ascii`
- `CARDDAV_ACTION ACTION`
- `time_t start`
- `time_t end`
- `char use_uri`

### 5.2.1 Detailed Description

A struct used to exchange all user input between various parts of the library.

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

- `carddav-utils.h`

## 5.3 `_response` Struct Reference

A struct used for returning messages from the library to users.

```
#include <carddav.h>
```

### Data Fields

- `char * msg`  
*String for storing response.*

### 5.3.1 Detailed Description

A struct used for returning messages from the library to users.

### 5.3.2 Field Documentation

#### 5.3.2.1 msg

```
char * _response::msg
```

String for storing response.

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

- [carddav.h](#)

## 5.4 config\_data Struct Reference

Used to exchange user options to the library.

```
#include <carddav-utils.h>
```

### Data Fields

- char **trace\_ascii**

#### 5.4.1 Detailed Description

Used to exchange user options to the library.

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

- carddav-utils.h

## 5.5 debug\_curl Struct Reference

### Data Fields

- int [trace\\_ascii](#)  
*0 or 1*
- int [debug](#)  
*0 or 1*
- int **verify\_ssl\_certificate**
- int **use\_locking**
- char \* **custom\_cacert**

### 5.5.1 Field Documentation

#### 5.5.1.1 debug

```
int debug_curl::debug
```

0 or 1

#### 5.5.1.2 trace\_ascii

```
int debug_curl::trace_ascii
```

0 or 1

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

- [carddav.h](#)

## 5.6 MD5\_CONTEXT Struct Reference

### Data Fields

- u32 **A**
- u32 **B**
- u32 **C**
- u32 **D**
- u32 **nblocks**
- unsigned char **buf** [64]
- int **count**
- int **finalized**

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

- md5.h

## 5.7 MemoryStruct Struct Reference

Used to hold messages between the CardDAV server and the library.

```
#include <carddav-utils.h>
```



## Data Fields

- `char * memory`
- `size_t size`

### 5.7.1 Detailed Description

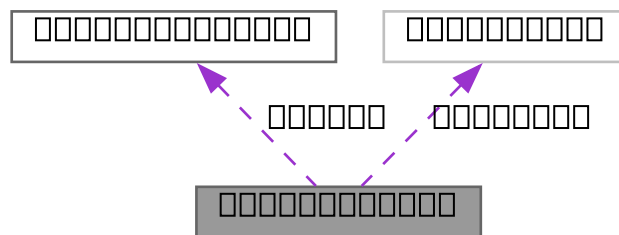
Used to hold messages between the CardDAV server and the library.

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

- `carddav-utils.h`

## 5.8 runtime\_info Struct Reference

Collaboration diagram for runtime\_info:



## Data Fields

- `carddav_error * error`
- `debug_curl * options`

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

- `carddav.h`



## Chapter 6

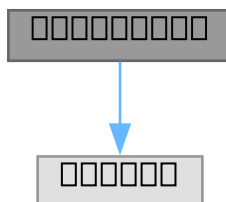
# File Documentation

### 6.1 carddav.h File Reference

interface to the carddav library.

```
#include <time.h>
```

Include dependency graph for carddav.h:



#### Data Structures

- struct [debug\\_curl](#)
- struct [\\_carddav\\_error](#)  
*A struct for storing error codes and messages.*
- struct [runtime\\_info](#)
- struct [\\_response](#)  
*A struct used for returning messages from the library to users.*

#### Macros

- `#define __CARDDAV_USERAGENT "libcurl-agent/0.1"`

## Typedefs

- typedef struct [\\_carddav\\_error](#) **carddav\_error**  
*Pointer to a [carddav\\_error](#) structure.*
- typedef struct [\\_response](#) **response**  
*Pointer to a [\\_response](#) structure.*

## Enumerations

- enum [CARDDAV\\_ACTION](#) {  
UNKNOWN , ADD , DELETE , FREEBUSY ,  
MODIFY , GET , GETALL , GETCALNAME ,  
ISCARDDAV , OPTIONS }  
*specifies supported CardDAV actions.*
- enum [CARDDAV\\_RESPONSE](#) {  
OK , FORBIDDEN , CONFLICT , LOCKED ,  
NOTIMPLEMENTED }  
*specifies CardDAV error states.*

## Functions

- [CARDDAV\\_RESPONSE](#) [carddav\\_add\\_object](#) (const char \*object, const char \*URL, [runtime\\_info](#) \*info)  
*Function for adding a new card.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_delete\\_object](#) (const char \*object, const char \*URL, [runtime\\_info](#) \*info)  
*Function for deleting a card.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_delete\\_object\\_by\\_uri](#) (const char \*object, const char \*URL, [runtime\\_info](#) \*info)  
*Function for deleting a card by URI.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_modify\\_object](#) (const char \*object, const char \*URL, [runtime\\_info](#) \*info)  
*Function for modifying a card.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_modify\\_object\\_by\\_uri](#) (const char \*object, const char \*URL, [runtime\\_info](#) \*info)  
*Function for modifying a card by URI.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_get\\_object](#) ([response](#) \*result, time\_t start, time\_t end, const char \*URL, [runtime\\_info](#) \*info)  
*Function for getting a collection of cards determined by time range.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_getall\\_object](#) ([response](#) \*result, const char \*URL, [runtime\\_info](#) \*info)  
*Function for getting all cards from the collection.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_getall\\_object\\_by\\_uri](#) ([response](#) \*result, const char \*URL, [runtime\\_info](#) \*info)  
*Function for getting all cards from the collection.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_get\\_displayname](#) ([response](#) \*result, const char \*URL, [runtime\\_info](#) \*info)  
*Function for getting the stored display name for the collection.*
- int [carddav\\_enabled\\_resource](#) (const char \*URL, [runtime\\_info](#) \*info)  
*Function to test whether a calendar resource is CardDAV enabled or not.*
- [CARDDAV\\_RESPONSE](#) [carddav\\_get\\_freebusy](#) ([response](#) \*result, time\_t start, time\_t end, const char \*URL, [runtime\\_info](#) \*info)  
*Function for getting free/busy information.*
- [carddav\\_error](#) \* [carddav\\_get\\_error](#) ([carddav\\_error](#) \*lib\_error)
- void [carddav\\_free\\_error](#) ([carddav\\_error](#) \*lib\_error)  
*Function for freeing memory for a previous initialization of a [carddav\\_error](#).*

- void `carddav_set_options` (`debug_curl` curl\_options)  
*Function which supports sending various options inside the library.*
- char \*\* `carddav_get_server_options` (const char \*URL, `runtime_info` \*info)  
*Function to call to get a list of supported CardDAV options for a server.*
- `runtime_info` \* `carddav_get_runtime_info` ()  
*Function for getting an initialized `runtime_info` structure.*
- void `carddav_free_runtime_info` (`runtime_info` \*\*info)  
*Function for freeing memory for a previous initialization of an info structure.*
- `response` \* `carddav_get_response` ()  
*Function for getting an initialized response structure.*
- void `carddav_free_response` (`response` \*\*info)  
*Function for freeing memory for a previous initialization of an response structure.*

### 6.1.1 Detailed Description

interface to the carddav library.

The library conforms to IETF carddav draft specification daboo-10. For further information follow this link <http://tools.ietf.org/html/draft-ietf-vcarddav-carddav-10>

### 6.1.2 Enumeration Type Documentation

#### 6.1.2.1 CARDDAV\_ACTION

enum `CARDDAV_ACTION`

specifies supported CardDAV actions.

UNKNOWN. An unknown action. ADD. Add a CardDAV calendar object. DELETE. Delete a CardDAV calendar object. MODIFY. Modify a CardDAV calendar object. GET. Get one or more CardDAV calendar object(s). GETALL. Get all CardDAV calendar objects.

#### 6.1.2.2 CARDDAV\_RESPONSE

enum `CARDDAV_RESPONSE`

specifies CardDAV error states.

OK (HTTP 200). Request was satisfied. FORBIDDEN (HTTP 403). Access not allowed. Dont repeat request. CONFLICT (HTTP 409). Conflict between current state of CardDAV collection and request. Client must solve the conflict and then resend request. LOCKED (HTTP 423). Locking failed.

### 6.1.3 Function Documentation

#### 6.1.3.1 carddav\_add\_object()

```
CARDDAV_RESPONSE carddav_add_object (
    const char * object,
    const char * URL,
    runtime_info * info )
```

Function for adding a new card.

## Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

## See also

[runtime\\_info](#)

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

Function for adding a new card.

## Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

6.1.3.2 `carddav_delete_object()`

```
CARDDAV_RESPONSE carddav_delete_object (
    const char * object,
    const char * URL,
    runtime_info * info )
```

Function for deleting a card.

## Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738). Generated by Doxygen
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

See also

[runtime\\_info](#)

Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

Function for deleting a card.

Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).

Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.3 carddav\_delete\_object\_by\_uri()

```
CARDDAV\_RESPONSE carddav_delete_object_by_uri (  
    const char * object,  
    const char * URL,  
    runtime\_info * info )
```

Function for deleting a card by URI.

Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

See also

[runtime\\_info](#)

**Returns**

Ok, FORBIDDEN, or CONFLICT.

**See also**

[CARDDAV\\_RESPONSE](#)

Function for deleting a card by URI.

**Parameters**

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).

**Returns**

Ok, FORBIDDEN, or CONFLICT.

**See also**

[CARDDAV\\_RESPONSE](#)

**6.1.3.4 carddav\_enabled\_resource()**

```
int carddav_enabled_resource (
    const char * URL,
    runtime_info * info )
```

Function to test whether a calendar resource is CardDAV enabled or not.

**Parameters**

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

**See also**

[runtime\\_info](#)

**Returns**

0 (zero) means no CardDAV support, otherwise CardDAV support detected.



## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
------------	---

## Returns

0 (zero) means no CardDAV support, otherwise CardDAV support detected.

### 6.1.3.5 carddav\_free\_error()

```
void carddav_free_error (
    carddav_error * lib_error )
```

Function for freeing memory for a previous initialization of a `carddav_error`.

## See also

[carddav\\_get\\_error\(\)](#) Caller provides a pointer to a local `carddav_error` structure.

## Parameters

<i>lib_error</i>	A pointer to a struct <code>_carddav_error</code> .
------------------	---

## See also

[\\_carddav\\_error](#)

### 6.1.3.6 carddav\_free\_response()

```
void carddav_free_response (
    response ** resp )
```

Function for freeing memory for a previous initialization of an response structure.

## Parameters

<i>info</i>	Address to a pointer to a response structure.
-------------	---

## See also

[\\_response](#)

### 6.1.3.7 carddav\_free\_runtime\_info()

```
void carddav_free_runtime_info (
    runtime_info ** info )
```

Function for freeing memory for a previous initialization of an info structure.

#### Parameters

<i>info</i>	Address to a pointer to a <a href="#">runtime_info</a> structure.
-------------	---

#### See also

[runtime\\_info](#)

### 6.1.3.8 carddav\_get\_displayname()

```
CARDDAV_RESPONSE carddav_get_displayname (
    response * result,
    const char * URL,
    runtime_info * info )
```

Function for getting the stored display name for the collection.

#### Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

#### See also

[response](#). Caller is responsible for freeing the memory.

#### Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

#### See also

[runtime\\_info](#)

#### Returns

Ok, FORBIDDEN, or CONFLICT.

#### See also

[CARDDAV\\_RESPONSE](#)

## Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

## See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
------------	--

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.9 carddav\_get\_error()

```
carddav_error * carddav_get_error (
    carddav_error * lib_error )
```

**Deprecated** Always returns an initialized empty `carddav_error` Function to call in case of errors. Caller provides a pointer to a local `carddav_error` structure. `Carddav_get_error` will initialize pointer if NULL. Caller is responsible for freeing returned memory. After the first call the internal error buffer is reset.

## Parameters

<i>lib_error</i>	A pointer to a struct <a href="#">_carddav_error</a> .
------------------	--

## See also

[\\_carddav\\_error](#)

## Returns

An initialized `carddav_error` pointer to memory where error messages can be found from the last call to the library.

**Deprecated** Function to call in case of errors. Caller provides a pointer to a local `carddav_error` structure. `Carddav_get_error` will initialize pointer if NULL. Caller is responsible for freeing returned memory. After the first call the internal error buffer is reset.

## Parameters

<i>lib_error</i>	A pointer to a struct <a href="#">_carddav_error</a> .
------------------	--

## See also

[\\_carddav\\_error](#)

## Returns

An initialized `carddav_error` pointer to memory where error messages can be found from the last call to the library.

**6.1.3.10 carddav\_get\_freebusy()**

```
CARDDAV_RESPONSE carddav_get_freebusy (
    response * result,
    time_t start,
    time_t end,
    const char * URL,
    runtime_info * info )
```

Function for getting free/busy information.

## Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

## See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>start</i>	<code>time_t</code> variable specifying start and end for range. Both are included in range.
<i>end</i>	<code>time_t</code> variable specifying start and end for range. Both are included in range.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.11 carddav\_get\_object()

```
CARDDAV_RESPONSE carddav_get_object (
    response * result,
    time_t start,
    time_t end,
    const char * URL,
    runtime_info * info )
```

Function for getting a collection of cards determined by time range.

#### Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

#### See also

[response](#). Caller is responsible for freeing the memory.

#### Parameters

<i>start</i>	time_t variable specifying start for range. Included in search.
<i>end</i>	time_t variable specifying end for range. Included in search.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

#### See also

[runtime\\_info](#)

#### Returns

Ok, FORBIDDEN, or CONFLICT.

#### See also

[CARDDAV\\_RESPONSE](#)

Function for getting a collection of cards determined by time range.

#### Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

#### See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>start</i>	time_t variable specifying start and end for range. Both are included in range.
<i>end</i>	time_t variable specifying start and end for range. Both are included in range.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.12 carddav\_get\_response()

```
response * carddav_get_response ( )
```

Function for getting an initialized response structure.

## Returns

response.

## See also

[\\_response](#)

### 6.1.3.13 carddav\_get\_runtime\_info()

```
runtime_info * carddav_get_runtime_info ( )
```

Function for getting an initialized [runtime\\_info](#) structure.

## Returns

[runtime\\_info](#).

## See also

[runtime\\_info](#)

### 6.1.3.14 carddav\_get\_server\_options()

```
char ** carddav_get_server_options (
    const char * URL,
    runtime_info * info )
```

Function to call to get a list of supported CardDAV options for a server.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

## See also

[runtime\\_info](#)

## Returns

A list of available options or NULL in case of any error.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
------------	--

## Returns

A list of available options or NULL in case of any error.

## 6.1.3.15 carddav\_getall\_object()

```
CARDDAV_RESPONSE carddav_getall_object (
    response * result,
    const char * URL,
    runtime_info * info )
```

Function for getting all cards from the collection.

## Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

## See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

See also

[runtime\\_info](#)

Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

Function for getting all cards from the collection.

Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

See also

[response](#). Caller is responsible for freeing the memory.

Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
------------	--

Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.16 `carddav_getall_object_by_uri()`

```
CARDDAV_RESPONSE carddav_getall_object_by_uri (
    response * result,
    const char * URL,
    runtime_info * info )
```

Function for getting all cards from the collection.

This version stores the URI as a VCARD parameter.



## Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

## See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

## See also

[runtime\\_info](#)

## Returns

Ok, FORBIDDEN, or CONFLICT.

## See also

[CARDDAV\\_RESPONSE](#)

Function for getting all cards from the collection.

This version stores the URI as a VCARD parameter.

## Parameters

<i>result</i>	A pointer to struct <a href="#">_response</a> where the result is to stored.
---------------	--

## See also

[response](#). Caller is responsible for freeing the memory.

## Parameters

<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
------------	--

## Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.17 carddav\_modify\_object()

```
CARDDAV_RESPONSE carddav_modify_object (
    const char * object,
    const char * URL,
    runtime_info * info )
```

Function for modifying a card.

#### Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

See also

[runtime\\_info](#)

#### Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

Function for modifying a card.

#### Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="#">http://</a> ][username[:password]@]host[:port]/url-path. See (RFC1738).

#### Returns

Ok, FORBIDDEN, or CONFLICT.

See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.18 carddav\_modify\_object\_by\_uri()

```
CARDDAV_RESPONSE carddav_modify_object_by_uri (
    const char * object,
    const char * URL,
    runtime_info * info )
```

Function for modifying a card by URI.

#### Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).
<i>info</i>	Pointer to a <a href="#">runtime_info</a> structure.

#### See also

[runtime\\_info](#)

#### Returns

Ok, FORBIDDEN, or CONFLICT.

#### See also

[CARDDAV\\_RESPONSE](#)

Function for modifying a card by URI.

#### Parameters

<i>object</i>	Appointment following ICal format (RFC2445). Receiver is responsible for freeing the memory.
<i>URL</i>	Defines CardDAV resource. Receiver is responsible for freeing the memory. [ <a href="http://[username[:password]@]host[:port]/url-path">http://[username[:password]@]host[:port]/url-path</a> . See (RFC1738).

#### Returns

Ok, FORBIDDEN, or CONFLICT.

#### See also

[CARDDAV\\_RESPONSE](#)

### 6.1.3.19 carddav\_set\_options()

```
void carddav_set_options (
    debug_curl curl_options )
```

Function which supports sending various options inside the library.

**Deprecated** Does nothing Function which supports sending various options inside the library.

#### Parameters

<i>curl_options</i>	A struct <a href="#">debug_curl</a> . See <a href="#">debug_curl</a> .
<i>curl_options</i>	A struct <a href="#">debug_curl</a> . See <a href="#">debug_curl</a> .

# Index

[\\_CARDDAV\\_SETTINGS](#), [10](#)

[\\_carddav\\_error](#), [9](#)

[str](#), [9](#)

[\\_response](#), [10](#)

[msg](#), [11](#)

[carddav.h](#), [15](#)

[CARDDAV\\_ACTION](#), [17](#)

[carddav\\_add\\_object](#), [17](#)

[carddav\\_delete\\_object](#), [18](#)

[carddav\\_delete\\_object\\_by\\_uri](#), [19](#)

[carddav\\_enabled\\_resource](#), [20](#)

[carddav\\_free\\_error](#), [21](#)

[carddav\\_free\\_response](#), [21](#)

[carddav\\_free\\_runtime\\_info](#), [21](#)

[carddav\\_get\\_displayname](#), [22](#)

[carddav\\_get\\_error](#), [23](#)

[carddav\\_get\\_freebusy](#), [24](#)

[carddav\\_get\\_object](#), [24](#)

[carddav\\_get\\_response](#), [26](#)

[carddav\\_get\\_runtime\\_info](#), [26](#)

[carddav\\_get\\_server\\_options](#), [26](#)

[carddav\\_getall\\_object](#), [27](#)

[carddav\\_getall\\_object\\_by\\_uri](#), [28](#)

[carddav\\_modify\\_object](#), [30](#)

[carddav\\_modify\\_object\\_by\\_uri](#), [30](#)

[CARDDAV\\_RESPONSE](#), [17](#)

[carddav\\_set\\_options](#), [31](#)

[CARDDAV\\_ACTION](#)

[carddav.h](#), [17](#)

[carddav\\_add\\_object](#)

[carddav.h](#), [17](#)

[carddav\\_delete\\_object](#)

[carddav.h](#), [18](#)

[carddav\\_delete\\_object\\_by\\_uri](#)

[carddav.h](#), [19](#)

[carddav\\_enabled\\_resource](#)

[carddav.h](#), [20](#)

[carddav\\_free\\_error](#)

[carddav.h](#), [21](#)

[carddav\\_free\\_response](#)

[carddav.h](#), [21](#)

[carddav\\_free\\_runtime\\_info](#)

[carddav.h](#), [21](#)

[carddav\\_get\\_displayname](#)

[carddav.h](#), [22](#)

[carddav\\_get\\_error](#)

[carddav.h](#), [23](#)

[carddav\\_get\\_freebusy](#)

[carddav.h](#), [24](#)

[carddav\\_get\\_object](#)

[carddav.h](#), [24](#)

[carddav\\_get\\_response](#)

[carddav.h](#), [26](#)

[carddav\\_get\\_runtime\\_info](#)

[carddav.h](#), [26](#)

[carddav\\_get\\_server\\_options](#)

[carddav.h](#), [26](#)

[carddav\\_getall\\_object](#)

[carddav.h](#), [27](#)

[carddav\\_getall\\_object\\_by\\_uri](#)

[carddav.h](#), [28](#)

[carddav\\_modify\\_object](#)

[carddav.h](#), [30](#)

[carddav\\_modify\\_object\\_by\\_uri](#)

[carddav.h](#), [30](#)

[CARDDAV\\_RESPONSE](#)

[carddav.h](#), [17](#)

[carddav\\_set\\_options](#)

[carddav.h](#), [31](#)

[config\\_data](#), [11](#)

[debug](#)

[debug\\_curl](#), [12](#)

[debug\\_curl](#), [11](#)

[debug](#), [12](#)

[trace\\_ascii](#), [12](#)

[MD5\\_CONTEXT](#), [12](#)

[MemoryStruct](#), [12](#)

[msg](#)

[\\_response](#), [11](#)

[runtime\\_info](#), [13](#)

[str](#)

[\\_carddav\\_error](#), [9](#)

[trace\\_ascii](#)

[debug\\_curl](#), [12](#)