

general\_ledger

Generated by Doxygen 1.8.1.2

Fri Jun 13 2014 19:53:09



# Contents

<b>1</b>	<b>General Ledger.</b>	<b>1</b>
<b>2</b>	<b>Module Index</b>	<b>3</b>
2.1	Modules . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class Hierarchy . . . . .	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class List . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Module Documentation</b>	<b>11</b>
6.1	Database interaction module . . . . .	11
6.1.1	Detailed Description . . . . .	11
6.1.2	Function Documentation . . . . .	12
6.1.2.1	get_connection . . . . .	12
6.1.2.2	get_database_type . . . . .	12
6.2	Program configuration module . . . . .	13
6.2.1	Detailed Description . . . . .	13
6.3	General purpose helpers. . . . .	14
6.3.1	Detailed Description . . . . .	14
6.3.2	Function Documentation . . . . .	14
6.3.2.1	split . . . . .	14
6.3.2.2	trim . . . . .	14
6.3.2.3	trim_back . . . . .	14
6.3.2.4	trim_front . . . . .	15
<b>7</b>	<b>Class Documentation</b>	<b>17</b>
7.1	genleg::Config Class Reference . . . . .	17
7.1.1	Detailed Description . . . . .	17
7.1.2	Constructor & Destructor Documentation . . . . .	17

7.1.2.1	Config	17
7.1.2.2	~Config	18
7.1.3	Member Function Documentation	18
7.1.3.1	add_cmdline_option	18
7.1.3.2	is_set	18
7.1.3.3	operator[]	18
7.1.3.4	populate_from_cmdline	18
7.1.3.5	populate_from_file	19
7.1.4	Member Data Documentation	19
7.1.4.1	m_opts_set	19
7.1.4.2	m_opts_supp	19
7.2	genleg::ConfigBadConfigFile Class Reference	19
7.2.1	Detailed Description	20
7.3	genleg::ConfigBadOption Class Reference	20
7.3.1	Detailed Description	21
7.4	genleg::ConfigCouldNotOpenFile Class Reference	21
7.4.1	Detailed Description	22
7.5	genleg::ConfigException Class Reference	22
7.5.1	Detailed Description	23
7.6	genleg::ConfigOptionNotSet Class Reference	23
7.6.1	Detailed Description	24
7.7	gldb::DBConn Class Reference	24
7.7.1	Detailed Description	25
7.7.2	Constructor & Destructor Documentation	25
7.7.2.1	DBConn	25
7.7.2.2	DBConn	25
7.7.3	Member Function Documentation	25
7.7.3.1	operator=	25
7.7.3.2	select	25
7.7.4	Member Data Documentation	25
7.7.4.1	m_imp	25
7.8	gldb::DBConnCouldNotConnect Class Reference	26
7.8.1	Detailed Description	26
7.8.2	Constructor & Destructor Documentation	26
7.8.2.1	DBConnCouldNotConnect	27
7.9	gldb::DBConnCouldNotQuery Class Reference	27
7.9.1	Detailed Description	28
7.9.2	Constructor & Destructor Documentation	28
7.9.2.1	DBConnCouldNotQuery	28
7.10	gldb::DBConnDummy Class Reference	28

7.10.1 Detailed Description . . . . .	29
7.10.2 Constructor & Destructor Documentation . . . . .	29
7.10.2.1 DBConnDummy . . . . .	29
7.10.2.2 DBConnDummy . . . . .	29
7.10.2.3 ~DBConnDummy . . . . .	29
7.10.3 Member Function Documentation . . . . .	29
7.10.3.1 operator= . . . . .	29
7.10.3.2 select . . . . .	29
7.11 glDb::DBConnException Class Reference . . . . .	30
7.11.1 Detailed Description . . . . .	30
7.11.2 Constructor & Destructor Documentation . . . . .	30
7.11.2.1 DBConnException . . . . .	30
7.12 glDb::DBConnImp Class Reference . . . . .	31
7.12.1 Detailed Description . . . . .	31
7.12.2 Constructor & Destructor Documentation . . . . .	31
7.12.2.1 DBConnImp . . . . .	31
7.12.2.2 ~DBConnImp . . . . .	31
7.12.3 Member Function Documentation . . . . .	31
7.12.3.1 select . . . . .	31
7.13 glDb::DBConnMySQL Class Reference . . . . .	32
7.13.1 Detailed Description . . . . .	33
7.13.2 Constructor & Destructor Documentation . . . . .	33
7.13.2.1 DBConnMySQL . . . . .	33
7.13.2.2 DBConnMySQL . . . . .	33
7.13.2.3 ~DBConnMySQL . . . . .	33
7.13.3 Member Function Documentation . . . . .	33
7.13.3.1 operator= . . . . .	33
7.13.3.2 select . . . . .	33
7.13.4 Member Data Documentation . . . . .	34
7.13.4.1 m_conn . . . . .	34
7.14 glDb::Table Class Reference . . . . .	34
7.14.1 Detailed Description . . . . .	35
7.14.2 Constructor & Destructor Documentation . . . . .	35
7.14.2.1 Table . . . . .	35
7.14.2.2 ~Table . . . . .	35
7.14.3 Member Function Documentation . . . . .	35
7.14.3.1 append_record . . . . .	35
7.14.3.2 get_headers . . . . .	35
7.14.3.3 num_fields . . . . .	36
7.14.3.4 num_records . . . . .	36

7.14.3.5	<a href="#">operator[]</a>	36
7.14.4	<a href="#">Member Data Documentation</a>	36
7.14.4.1	<a href="#">m_headers</a>	36
7.14.4.2	<a href="#">m_records</a>	36
7.15	<a href="#">gldb::TableField Class Reference</a>	36
7.15.1	<a href="#">Detailed Description</a>	38
7.15.2	<a href="#">Constructor &amp; Destructor Documentation</a>	38
7.15.2.1	<a href="#">TableField</a>	38
7.15.2.2	<a href="#">TableField</a>	38
7.15.2.3	<a href="#">~TableField</a>	38
7.15.3	<a href="#">Member Function Documentation</a>	38
7.15.3.1	<a href="#">length</a>	38
7.15.3.2	<a href="#">operator std::string</a>	38
7.15.3.3	<a href="#">operator+=</a>	38
7.15.3.4	<a href="#">operator+=</a>	39
7.15.3.5	<a href="#">operator=</a>	39
7.15.3.6	<a href="#">operator=</a>	39
7.15.3.7	<a href="#">operator[]</a>	39
7.15.3.8	<a href="#">operator[]</a>	40
7.15.4	<a href="#">Friends And Related Function Documentation</a>	40
7.15.4.1	<a href="#">operator&lt;&lt;</a>	40
7.15.5	<a href="#">Member Data Documentation</a>	40
7.15.5.1	<a href="#">m_data</a>	40
7.16	<a href="#">gldb::TableRow Class Reference</a>	40
7.16.1	<a href="#">Detailed Description</a>	41
7.16.2	<a href="#">Constructor &amp; Destructor Documentation</a>	41
7.16.2.1	<a href="#">TableRow</a>	41
7.16.2.2	<a href="#">TableRow</a>	41
7.16.2.3	<a href="#">~TableRow</a>	41
7.16.3	<a href="#">Member Function Documentation</a>	41
7.16.3.1	<a href="#">append_field</a>	41
7.16.3.2	<a href="#">append_field</a>	41
7.16.3.3	<a href="#">append_field</a>	42
7.16.3.4	<a href="#">operator[]</a>	42
7.16.3.5	<a href="#">operator[]</a>	42
7.16.3.6	<a href="#">print</a>	42
7.16.3.7	<a href="#">size</a>	42
7.16.4	<a href="#">Member Data Documentation</a>	43
7.16.4.1	<a href="#">m_fields</a>	43

<b>8 File Documentation</b>	<b>45</b>
8.1 lib/config/config.cpp File Reference	45
8.1.1 Detailed Description	45
8.2 lib/config/config.h File Reference	46
8.2.1 Detailed Description	47
8.3 lib/config/config_getopt.cpp File Reference	47
8.3.1 Detailed Description	47
8.3.2 Macro Definition Documentation	48
8.3.2.1 _XOPEN_SOURCE	48
8.4 lib/database/data_structures.h File Reference	48
8.4.1 Detailed Description	49
8.5 lib/database/database.h File Reference	50
8.5.1 Detailed Description	51
8.6 lib/database/dbconn.cpp File Reference	51
8.6.1 Detailed Description	52
8.7 lib/database/dbconn.h File Reference	53
8.7.1 Detailed Description	54
8.8 lib/database/dbconnimp.h File Reference	54
8.8.1 Detailed Description	56
8.9 lib/database/table.cpp File Reference	57
8.9.1 Detailed Description	57
8.10 lib/database/table.h File Reference	58
8.10.1 Detailed Description	59
8.11 lib/database/tablefield.cpp File Reference	59
8.11.1 Detailed Description	60
8.12 lib/database/tablefield.h File Reference	60
8.12.1 Detailed Description	62
8.13 lib/database/ablerow.cpp File Reference	62
8.13.1 Detailed Description	62
8.14 lib/database/ablerow.h File Reference	63
8.14.1 Detailed Description	64
8.15 lib/database_imp/database_imp.h File Reference	64
8.15.1 Detailed Description	65
8.16 lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference	66
8.16.1 Detailed Description	67
8.17 lib/database_imp/dummy/dbconn_dummy_imp.h File Reference	68
8.17.1 Detailed Description	69
8.18 lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference	70
8.18.1 Detailed Description	70
8.19 lib/database_imp/mysql/dbconn_mysql_imp.h File Reference	71

8.19.1 Detailed Description . . . . .	72
8.20 lib/stringhelp/stringhelp.cpp File Reference . . . . .	72
8.20.1 Detailed Description . . . . .	72
8.21 lib/stringhelp/stringhelp.h File Reference . . . . .	73
8.21.1 Detailed Description . . . . .	73



## Chapter 1

# General Ledger.

General Ledger will be a fully-featured, multi-user, open-source general ledger system. The project is in the early stages of development.



## Chapter 2

# Module Index

### 2.1 Modules

Here is a list of all modules:

Database interaction module . . . . .	11
Program configuration module . . . . .	13
General purpose helpers. . . . .	14



## Chapter 3

# Class Index

### 3.1 Class Hierarchy

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

genleg::Config . . . . .	17
genleg::ConfigException . . . . .	22
genleg::ConfigBadConfigFile . . . . .	19
genleg::ConfigBadOption . . . . .	20
genleg::ConfigCouldNotOpenFile . . . . .	21
genleg::ConfigOptionNotSet . . . . .	23
gldb::DBConn . . . . .	24
gldb::DBConnException . . . . .	30
gldb::DBConnCouldNotConnect . . . . .	26
gldb::DBConnCouldNotQuery . . . . .	27
gldb::DBConnImp . . . . .	31
gldb::DBConnDummy . . . . .	28
gldb::DBConnMySQL . . . . .	32
gldb::Table . . . . .	34
gldb::TableField . . . . .	36
gldb::TableRow . . . . .	40



## Chapter 4

# Class Index

### 4.1 Class List

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

<a href="#">genleg::Config</a>	Configuration options class . . . . .	17
<a href="#">genleg::ConfigBadConfigFile</a>	Exception class for badly formed configuration file . . . . .	19
<a href="#">genleg::ConfigBadOption</a>	Exception class for bad provided option . . . . .	20
<a href="#">genleg::ConfigCouldNotOpenFile</a>	Exception class for when conf file cannot be opened . . . . .	21
<a href="#">genleg::ConfigException</a>	Configuration module exception base class . . . . .	22
<a href="#">genleg::ConfigOptionNotSet</a>	Exception class for option not set . . . . .	23
<a href="#">gldb::DBConn</a>	Database connection class . . . . .	24
<a href="#">gldb::DBConnCouldNotConnect</a>	Could not connect to database exception class . . . . .	26
<a href="#">gldb::DBConnCouldNotQuery</a>	Could not execute database query exception class . . . . .	27
<a href="#">gldb::DBConnDummy</a>	Dummy database implementation class . . . . .	28
<a href="#">gldb::DBConnException</a>	Base database connection exception class . . . . .	30
<a href="#">gldb::DBConnImp</a>	Abstract database implementation base class . . . . .	31
<a href="#">gldb::DBConnMySQL</a>	MySQL database implementation class . . . . .	32
<a href="#">gldb::Table</a>	Database table class . . . . .	34
<a href="#">gldb::TableField</a>	Database table field class . . . . .	36
<a href="#">gldb::TableRow</a>	Database table row class . . . . .	40





## Chapter 5

# File Index

### 5.1 File List

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

lib/config/ <a href="#">config.cpp</a>	
Implementation of program configurations class . . . . .	45
lib/config/ <a href="#">config.h</a>	
Interface to program configurations class . . . . .	46
lib/config/ <a href="#">config_getopt.cpp</a>	
Implementation of command line functionality . . . . .	47
lib/database/ <a href="#">data_structures.h</a>	
Main interface to database data structures . . . . .	48
lib/database/ <a href="#">database.h</a>	
User interface to database functionality . . . . .	50
lib/database/ <a href="#">dbconn.cpp</a>	
Implementation of database connection class . . . . .	51
lib/database/ <a href="#">dbconn.h</a>	
Interface to database connection base class . . . . .	53
lib/database/ <a href="#">dbconnimp.h</a>	
Interface to abstract database implementation base class . . . . .	54
lib/database/ <a href="#">table.cpp</a>	
Implementation of database table data structure . . . . .	57
lib/database/ <a href="#">table.h</a>	
Interface to database table data structure . . . . .	58
lib/database/ <a href="#">tablefield.cpp</a>	
Implementation of database table field class . . . . .	59
lib/database/ <a href="#">tablefield.h</a>	
Interface to database table field class . . . . .	60
lib/database/ <a href="#">tablerow.cpp</a>	
Implementation of database table row data structure . . . . .	62
lib/database/ <a href="#">tablerow.h</a>	
Interface to database table row data structure . . . . .	63
lib/database_imp/ <a href="#">database_imp.h</a>	
Interface to database implementation factory function . . . . .	64
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.cpp</a>	
Implementation of Dummy database connection implementation class . . . . .	66
lib/database_imp/dummy/ <a href="#">dbconn_dummy_imp.h</a>	
Interface to dummy database connection implementation class . . . . .	68
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.cpp</a>	
Implementation of MySQL database connection implementation class . . . . .	70
lib/database_imp/mysql/ <a href="#">dbconn_mysql_imp.h</a>	
Interface to MySQL database connection implementation class . . . . .	71

lib/stringhelp/ <a href="#">stringhelp.cpp</a>	
Implementation of string helper functions . . . . .	72
lib/stringhelp/ <a href="#">stringhelp.h</a>	
Interface to string helper functions . . . . .	73

## Chapter 6

# Module Documentation

### 6.1 Database interaction module

#### Classes

- class [gldb::DBConnException](#)  
*Base database connection exception class.*
- class [gldb::DBConnCouldNotConnect](#)  
*Could not connect to database exception class.*
- class [gldb::DBConnCouldNotQuery](#)  
*Could not execute database query exception class.*
- class [gldb::DBConn](#)  
*Database connection class.*
- class [gldb::DBConnImp](#)  
*Abstract database implementation base class.*
- class [gldb::Table](#)  
*Database table class.*
- class [gldb::TableField](#)  
*Database table field class.*
- class [gldb::TableRow](#)  
*Database table row class.*
- class [gldb::DBConnDummy](#)  
*Dummy database implementation class.*
- class [gldb::DBConnMySQL](#)  
*MySQL database implementation class.*

#### Functions

- [DBConnImp \\*](#) [gldb::get\\_connection](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Creates and returns a pointer to a database implementation.*
- std::string [gldb::get\\_database\\_type](#) ()  
*Returns the name of the compiled-in database type.*

#### 6.1.1 Detailed Description

Module for interacting with the database.

## 6.1.2 Function Documentation

### 6.1.2.1 DBConnImp \* glldb::get\_connection ( const std::string *database*, const std::string *hostname*, const std::string *username*, const std::string *password* )

Creates and returns a pointer to a database implementation.

The implementation of this function is provided by the individual database implementations. One database implementation is compiled into the program at any one time. Multiple database systems are, or will be, supported, and not every system will possess the libraries and headers to compile every implementation. Therefore, only one implementation is compiled in at a time. The fact that each database implementation will implement this function to return the correct derived class prevents any attempt to compile unsupported library code. This would not be feasible if we were to simply provide each implementation as a subclass.

#### Parameters

<i>database</i>	The name of the database to which to connect.
<i>hostname</i>	The hostname of the computer running the database.
<i>username</i>	The username with which to log into the database.
<i>password</i>	The password with which to log into the database.

#### Returns

A pointer to the database implementation.

### 6.1.2.2 std::string glldb::get\_database\_type ( )

Returns the name of the compiled-in database type.

#### Returns

The name of the compiled-in database type.

## 6.2 Program configuration module

### Classes

- class [genleg::ConfigException](#)  
*Configuration module exception base class.*
- class [genleg::ConfigOptionNotSet](#)  
*Exception class for option not set.*
- class [genleg::ConfigBadOption](#)  
*Exception class for bad provided option.*
- class [genleg::ConfigCouldNotOpenFile](#)  
*Exception class for when conf file cannot be opened.*
- class [genleg::ConfigBadConfigFile](#)  
*Exception class for badly formed configuration file.*
- class [genleg::Config](#)  
*Configuration options class.*

### Enumerations

- enum [genleg::Argument](#)  
*Enumeration class for option argument specifications.*

#### 6.2.1 Detailed Description

Module for getting options from the command line and configuration files.

## 6.3 General purpose helpers.

### Functions

- `std::string & pgstring::trim_front (std::string &s)`  
*Trims leading whitespace from a string.*
- `std::string & pgstring::trim_back (std::string &s)`  
*Trims trailing whitespace from a string.*
- `std::string & pgstring::trim (std::string &s)`  
*Trims leading and trailing whitespace from a string.*
- `std::vector< std::string > pgstring::split (const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*

### 6.3.1 Detailed Description

General purpose helper classes and functions.

### 6.3.2 Function Documentation

#### 6.3.2.1 `std::vector< std::string > pgstring::split ( const std::string & s, const char delim )`

Splits a delimited string into tokens.

##### Parameters

<code>s</code>	The string to split.
<code>delim</code>	The delimiter character on which to split.

##### Returns

A vector of tokens.

#### 6.3.2.2 `std::string & pgstring::trim ( std::string & s )`

Trims leading and trailing whitespace from a string.

##### Parameters

<code>s</code>	The string to trim.
----------------	---------------------

##### Returns

The trimmed string.

#### 6.3.2.3 `std::string & pgstring::trim_back ( std::string & s )`

Trims trailing whitespace from a string.

##### Parameters

<code>s</code>	The string to trim.
----------------	---------------------

**Returns**

The trimmed string.

**6.3.2.4 `std::string & pgstring::trim_front ( std::string & s )`**

Trims leading whitespace from a string.

**Parameters**

<b>s</b>	The string to trim.
----------	---------------------

**Returns**

The trimmed string.





## Chapter 7

# Class Documentation

### 7.1 genleg::Config Class Reference

Configuration options class.

```
#include <config.h>
```

#### Public Member Functions

- [Config](#) ()
- [~Config](#) ()
- void [add\\_cmdline\\_option](#) (const std::string option, const enum [Argument](#) arg)  
*Adds a supported command line option.*
- void [populate\\_from\\_cmdline](#) (const int argc, char \*const \*argv)  
*Populates options from the command line.*
- void [populate\\_from\\_file](#) (const std::string filename)  
*Populates options from a configuration file.*
- bool [is\\_set](#) (const std::string option) const  
*Checks if an option is set.*
- const std::string & [operator\[\]](#) (const std::string &option) const  
*operator[] overload.*

#### Private Attributes

- std::map< std::string,  
std::string > [m\\_opts\\_set](#)
- std::list< std::pair  
< std::string, enum [Argument](#) > > [m\\_opts\\_supp](#)

#### 7.1.1 Detailed Description

Configuration options class.

#### 7.1.2 Constructor & Destructor Documentation

##### 7.1.2.1 Config::Config ( )

Constructor

### 7.1.2.2 Config::~Config ( )

Destructor

## 7.1.3 Member Function Documentation

### 7.1.3.1 void Config::add\_cmdline\_option ( const std::string *option*, const enum Argument *arg* )

Adds a supported command line option.

#### Parameters

<i>option</i>	The name of the option.
<i>arg</i>	The argument specification for the option.

### 7.1.3.2 bool Config::is\_set ( const std::string *option* ) const

Checks is an option is set.

#### Parameters

<i>option</i>	The name of the option to check.
---------------	----------------------------------

#### Returns

`true` if the option has been set, `false` if it has not.

### 7.1.3.3 const std::string & Config::operator[] ( const std::string & *option* ) const

operator[] overload.

Retrieves the value of a set option.

#### Parameters

<i>option</i>	The name of the option.
---------------	-------------------------

#### Returns

The value of the option.

#### Exceptions

<a href="#"><i>ConfigOptionNotSet</i></a>	If the named option has not been set.
---	---------------------------------------

### 7.1.3.4 void Config::populate\_from\_cmdline ( const int *argc*, char \*const \* *argv* )

Populates options from the command line.

#### Parameters

<i>argc</i>	<i>argc</i> supplied to <code>main()</code> .
<i>argv</i>	<i>argv</i> supplied to <code>main()</code> .

## Exceptions

<a href="#"><i>ConfigBadOption</i></a>	If an unsupported option is specified, or if a required argument is missing, or if an unexpected argument is found.
--	---

## 7.1.3.5 void Config::populate\_from\_file ( const std::string filename )

Populates options from a configuration file.

## Parameters

<i>filename</i>	The name of the configuration file.
-----------------	-------------------------------------

## Exceptions

<a href="#"><i>ConfigCouldNotOpenFile</i></a>	If the configuration file cannot be opened.
<a href="#"><i>ConfigBadConfigFile</i></a>	If the configuration file is badly formed.

## 7.1.4 Member Data Documentation

## 7.1.4.1 std::map&lt;std::string, std::string&gt; genleg::Config::m\_opts\_set [private]

Map of options which have been set

## 7.1.4.2 std::list&lt;std::pair&lt;std::string, enum Argument&gt; &gt; genleg::Config::m\_opts\_supp [private]

List of options which are supported

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

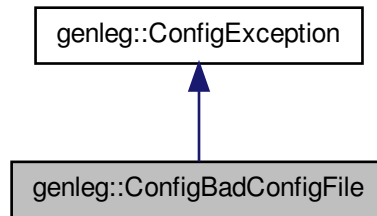
- lib/config/[config.h](#)
- lib/config/[config.cpp](#)
- lib/config/[config\\_getopt.cpp](#)

## 7.2 genleg::ConfigBadConfigFile Class Reference

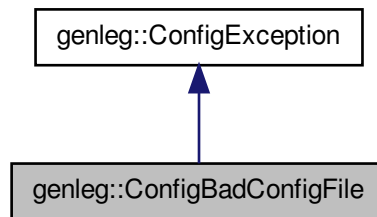
Exception class for badly formed configuration file.

```
#include <config.h>
```

Inheritance diagram for `genleg::ConfigBadConfigFile`:



Collaboration diagram for `genleg::ConfigBadConfigFile`:



### 7.2.1 Detailed Description

Exception class for badly formed configuration file.

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

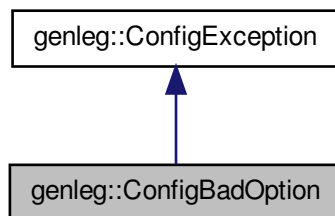
- `lib/config/config.h`

## 7.3 `genleg::ConfigBadOption` Class Reference

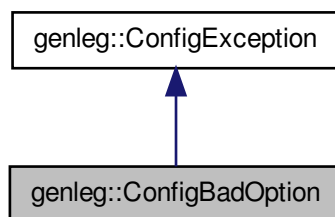
Exception class for bad provided option.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigBadOption:



Collaboration diagram for genleg::ConfigBadOption:



### 7.3.1 Detailed Description

Exception class for bad provided option.

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

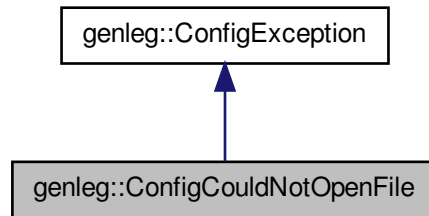
- lib/config/[config.h](#)

## 7.4 genleg::ConfigCouldNotOpenFile Class Reference

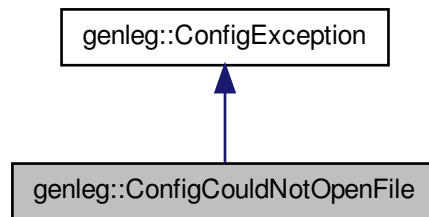
Exception class for when conf file cannot be opened.

```
#include <config.h>
```

Inheritance diagram for `genleg::ConfigCouldNotOpenFile`:



Collaboration diagram for `genleg::ConfigCouldNotOpenFile`:



#### 7.4.1 Detailed Description

Exception class for when conf file cannot be opened.

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

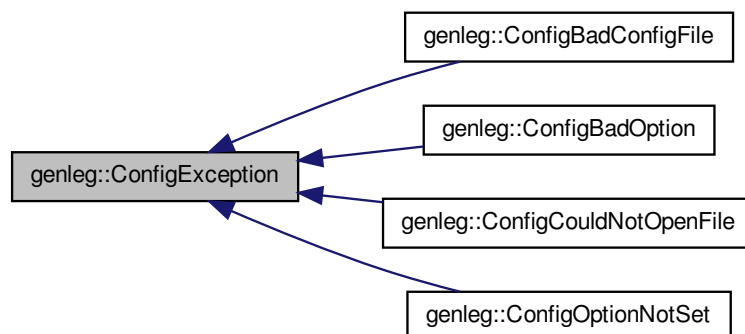
- `lib/config/config.h`

### 7.5 `genleg::ConfigException` Class Reference

Configuration module exception base class.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigException:



### 7.5.1 Detailed Description

Configuration module exception base class.

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

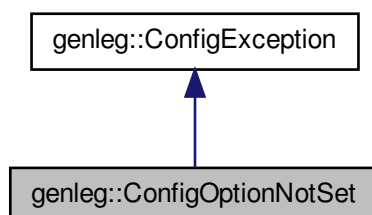
- `lib/config/config.h`

## 7.6 genleg::ConfigOptionNotSet Class Reference

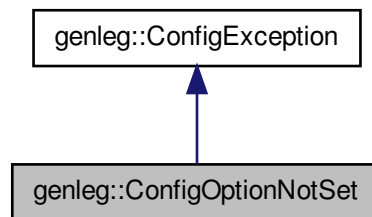
Exception class for option not set.

```
#include <config.h>
```

Inheritance diagram for genleg::ConfigOptionNotSet:



Collaboration diagram for genleg::ConfigOptionNotSet:



### 7.6.1 Detailed Description

Exception class for option not set.

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

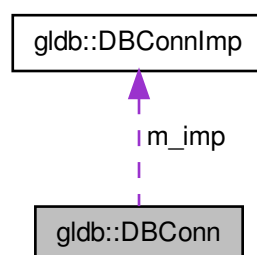
- `lib/config/config.h`

## 7.7 glldb::DBConn Class Reference

Database connection class.

```
#include <dbconn.h>
```

Collaboration diagram for glldb::DBConn:



### Public Member Functions

- `DBConn (DBConnImp *imp)`  
*Constructor.*
- `~DBConn ()`  
*Destructor..*



- [Table select](#) (std::string query)  
*Runs an SQL SELECT query.*
- [DBConn](#) (const [DBConn](#) &)
- [DBConn](#) & [operator=](#) (const [DBConn](#) &)

### Private Attributes

- [DBConnImp](#) \* [m\\_imp](#)

### 7.7.1 Detailed Description

Database connection class.

### 7.7.2 Constructor & Destructor Documentation

#### 7.7.2.1 DBConn::DBConn ( DBConnImp \* imp ) [explicit]

Constructor.

##### Parameters

<i>imp</i>	Pointer to database implementation object.
------------	--

#### 7.7.2.2 gldb::DBConn::DBConn ( const DBConn & )

Deleted copy constructor

### 7.7.3 Member Function Documentation

#### 7.7.3.1 DBConn& gldb::DBConn::operator= ( const DBConn & )

Deleted assignment operator

#### 7.7.3.2 Table DBConn::select ( std::string query )

Runs an SQL SELECT query.

##### Parameters

<i>query</i>	The query.
--------------	------------

##### Returns

A [Table](#) object containing the results.

### 7.7.4 Member Data Documentation

#### 7.7.4.1 DBConnImp\* gldb::DBConn::m\_imp [private]

Pointer to database implementation object.

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

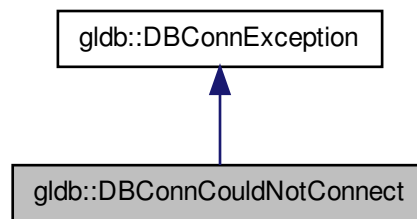
- [lib/database/dbconn.h](#)
- [lib/database/dbconn.cpp](#)

## 7.8 glldb::DBConnCouldNotConnect Class Reference

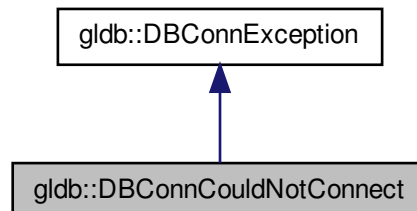
Could not connect to database exception class.

```
#include <dbconn.h>
```

Inheritance diagram for glldb::DBConnCouldNotConnect:



Collaboration diagram for glldb::DBConnCouldNotConnect:



### Public Member Functions

- [DBConnCouldNotConnect](#) (const std::string &msg)  
*Constructor.*

#### 7.8.1 Detailed Description

Could not connect to database exception class.

#### 7.8.2 Constructor & Destructor Documentation

7.8.2.1 `glldb::DBConnCouldNotConnect::DBConnCouldNotConnect ( const std::string & msg ) [inline], [explicit]`

Constructor.

Parameters

<i>msg</i>	Database error message
------------	------------------------

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

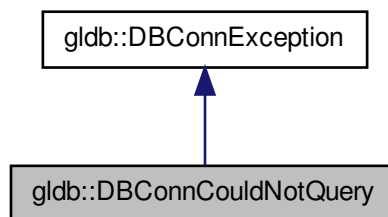
- [lib/database/dbconn.h](#)

## 7.9 glldb::DBConnCouldNotQuery Class Reference

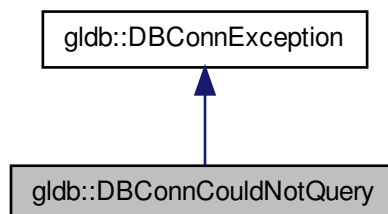
Could not execute database query exception class.

```
#include <dbconn.h>
```

Inheritance diagram for `glldb::DBConnCouldNotQuery`:



Collaboration diagram for `glldb::DBConnCouldNotQuery`:



### Public Member Functions

- [DBConnCouldNotQuery](#) (const std::string &msg)  
*Constructor.*

### 7.9.1 Detailed Description

Could not execute database query exception class.

### 7.9.2 Constructor & Destructor Documentation

7.9.2.1 `gldb::DBConnCouldNotQuery::DBConnCouldNotQuery ( const std::string & msg ) [inline], [explicit]`

Constructor.

#### Parameters

<i>msg</i>	Database error message
------------	------------------------

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

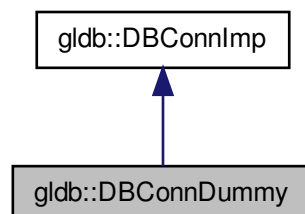
- [lib/database/dbconn.h](#)

## 7.10 gldb::DBConnDummy Class Reference

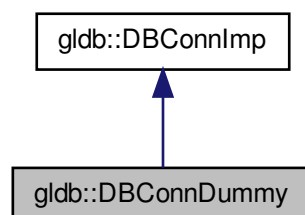
Dummy database implementation class.

```
#include <dbconn_dummy_imp.h>
```

Inheritance diagram for `gldb::DBConnDummy`:



Collaboration diagram for `gldb::DBConnDummy`:



## Public Member Functions

- [DBConnDummy](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Constructor.*
- [DBConnDummy](#) (const [DBConnDummy](#) &)
- virtual [~DBConnDummy](#) ()
- [DBConnDummy](#) & [operator=](#) (const [DBConnDummy](#) &)
- [Table select](#) (std::string query)  
*Fakes running of an SQL SELECT query.*

### 7.10.1 Detailed Description

Dummy database implementation class.

### 7.10.2 Constructor & Destructor Documentation

#### 7.10.2.1 DBConnDummy::DBConnDummy ( const std::string database, const std::string hostname, const std::string username, const std::string password )

Constructor.

##### Parameters

<i>database</i>	The name of the Dummy database.
<i>hostname</i>	The hostname of the server.
<i>username</i>	The username to log into the database.
<i>password</i>	The password to log into the database.

#### 7.10.2.2 glldb::DBConnDummy::DBConnDummy ( const DBConnDummy & )

Deleted copy constructor

#### 7.10.2.3 DBConnDummy::~~DBConnDummy ( ) [virtual]

Destructor

### 7.10.3 Member Function Documentation

#### 7.10.3.1 DBConnDummy& glldb::DBConnDummy::operator= ( const DBConnDummy & )

Deleted assignment operator

#### 7.10.3.2 Table DBConnDummy::select ( std::string query ) [virtual]

Fakes running of an SQL SELECT query.

##### Parameters

<i>query</i>	Any query.
--------------	------------

**Returns**

A [Table](#) object containing dummy results.

Implements [gldb::DBConnImp](#).

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

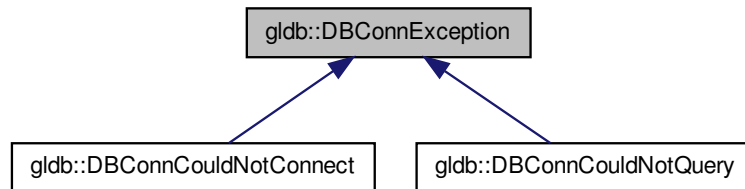
- [lib/database\\_imp/dummy/dbconn\\_dummy\\_imp.h](#)
- [lib/database\\_imp/dummy/dbconn\\_dummy\\_imp.cpp](#)

## 7.11 gldb::DBConnException Class Reference

Base database connection exception class.

```
#include <dbconn.h>
```

Inheritance diagram for gldb::DBConnException:

**Public Member Functions**

- [DBConnException](#) (const std::string &msg)  
*Constructor.*

### 7.11.1 Detailed Description

Base database connection exception class.

### 7.11.2 Constructor & Destructor Documentation

#### 7.11.2.1 gldb::DBConnException::DBConnException ( const std::string & msg ) [inline],[explicit]

Constructor.

**Parameters**

<i>msg</i>	Database error message
------------	------------------------

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

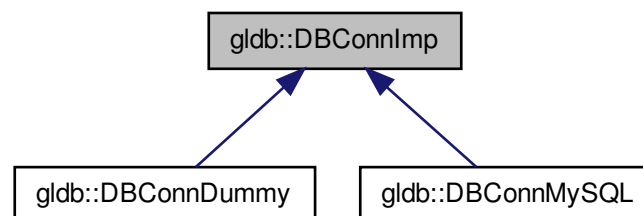
- [lib/database/dbconn.h](#)

## 7.12 gldb::DBConnImp Class Reference

Abstract database implementation base class.

```
#include <dbconnimp.h>
```

Inheritance diagram for gldb::DBConnImp:



### Public Member Functions

- [DBConnImp](#) ()
- virtual [~DBConnImp](#) ()
- virtual [Table select](#) (std::string query)=0  
*Runs an SQL SELECT query.*

### 7.12.1 Detailed Description

Abstract database implementation base class.

### 7.12.2 Constructor & Destructor Documentation

7.12.2.1 `gldb::DBConnImp::DBConnImp ( ) [inline]`

Constructor

7.12.2.2 `virtual gldb::DBConnImp::~~DBConnImp ( ) [inline],[virtual]`

Destructor

### 7.12.3 Member Function Documentation

7.12.3.1 `virtual Table gldb::DBConnImp::select ( std::string query ) [pure virtual]`

Runs an SQL SELECT query.

Parameters

<i>query</i>	The query.
--------------	------------

**Returns**

A [Table](#) object containing the results.

Implemented in [gldb::DBConnMySQL](#), and [gldb::DBConnDummy](#).

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

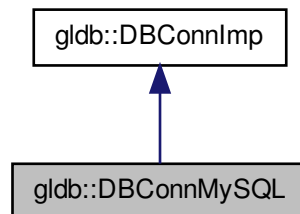
- lib/database/[dbconnimp.h](#)

## 7.13 gldb::DBConnMySQL Class Reference

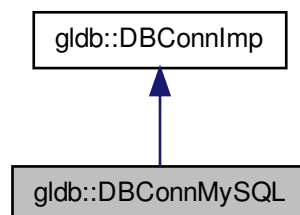
MySQL database implementation class.

```
#include <dbconn_mysql_imp.h>
```

Inheritance diagram for gldb::DBConnMySQL:



Collaboration diagram for gldb::DBConnMySQL:

**Public Member Functions**

- [DBConnMySQL](#) (const std::string database, const std::string hostname, const std::string username, const std::string password)

*Constructor.*

- [DBConnMySQL](#) (const [DBConnMySQL](#) &)



- virtual [~DBConnMySQL](#) ()
- [DBConnMySQL](#) & [operator=](#) (const [DBConnMySQL](#) &)
- [Table select](#) (std::string query)  
Runs an SQL SELECT query.

### Private Attributes

- MYSQL \* [m\\_conn](#)

### 7.13.1 Detailed Description

MySQL database implementation class.

### 7.13.2 Constructor & Destructor Documentation

- 7.13.2.1 [DBConnMySQL::DBConnMySQL](#) ( const std::string *database*, const std::string *hostname*, const std::string *username*, const std::string *password* )

Constructor.

#### Parameters

<i>database</i>	The name of the MySQL database.
<i>hostname</i>	The hostname of the server.
<i>username</i>	The username to log into the database.
<i>password</i>	The password to log into the database.

#### Exceptions

<a href="#">DBConnCouldNotConnect</a>	If could not connect to database.
---------------------------------------	-----------------------------------

- 7.13.2.2 [gldb::DBConnMySQL::DBConnMySQL](#) ( const [DBConnMySQL](#) & )

Deleted copy constructor

- 7.13.2.3 [DBConnMySQL::~~DBConnMySQL](#) ( ) [[virtual](#)]

Destructor

### 7.13.3 Member Function Documentation

- 7.13.3.1 [DBConnMySQL& gldb::DBConnMySQL::operator=](#) ( const [DBConnMySQL](#) & )

Deleted assignment operator

- 7.13.3.2 [Table DBConnMySQL::select](#) ( std::string *query* ) [[virtual](#)]

Runs an SQL SELECT query.

## Parameters

<i>query</i>	The query.
--------------	------------

## Returns

A [Table](#) object containing the results.

## Exceptions

<a href="#">DBConnCouldNotQuery</a>	If could not successfully execute query.
-------------------------------------	--

Implements [gldb::DBConnImp](#).

### 7.13.4 Member Data Documentation

#### 7.13.4.1 MySQL\* gldb::DBConnMySQL::m\_conn [private]

The initialized MySQL handle.

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

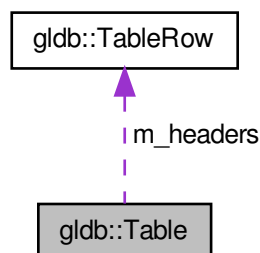
- lib/database\_imp/mysql/dbconn\_mysql\_imp.h
- lib/database\_imp/mysql/dbconn\_mysql\_imp.cpp

## 7.14 gldb::Table Class Reference

Database table class.

```
#include <table.h>
```

Collaboration diagram for gldb::Table:



### Public Member Functions

- [Table](#) (const [TableRow](#) &headers)  
*Constructor.*
- [~Table](#) ()
- [size\\_t num\\_fields](#) () const

- Returns the number of fields in each row.*
- `size_t num_records () const`  
*Returns the number of record in the table.*
- `const TableRow & get_headers () const`  
*Returns the field names.*
- `const TableRow & operator[] (const size_t idx) const`  
*Overloaded index operator.*
- `void append_record (const TableRow &new_record)`  
*Appends a record to the table.*

### Private Attributes

- `TableRow m_headers`
- `std::vector< TableRow > m_records`

## 7.14.1 Detailed Description

Database table class.

## 7.14.2 Constructor & Destructor Documentation

### 7.14.2.1 Table::Table ( const TableRow & headers ) [explicit]

Constructor.

Parameters

<i>headers</i>	Table row containing field names.
----------------	-----------------------------------

### 7.14.2.2 Table::~~Table ( )

Destructor

## 7.14.3 Member Function Documentation

### 7.14.3.1 void Table::append\_record ( const TableRow & new\_record )

Appends a record to the table.

Parameters

<i>new_record</i>	The record to append.
-------------------	-----------------------

### 7.14.3.2 const TableRow & Table::get\_headers ( ) const

Returns the field names.

Returns

The field names.

#### 7.14.3.3 `size_t Table::num_fields ( ) const`

Returns the number of fields in each row.

##### Returns

The number of fields in each row.

#### 7.14.3.4 `size_t Table::num_records ( ) const`

Returns the number of record in the table.

##### Returns

The number of records in the table.

#### 7.14.3.5 `const TableRow & Table::operator[] ( const size_t idx ) const`

Overloaded index operator.

##### Parameters

<code>idx</code>	The zero-based index of the record.
------------------	-------------------------------------

##### Returns

The selected record.

### 7.14.4 Member Data Documentation

#### 7.14.4.1 `TableRow glldb::Table::m_headers` `[private]`

The names of the fields

#### 7.14.4.2 `std::vector<TableRow> glldb::Table::m_records` `[private]`

A vector of the records

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

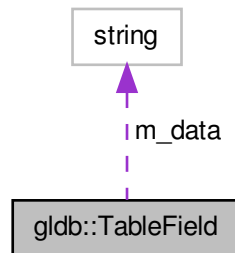
- [lib/database/table.h](#)
- [lib/database/table.cpp](#)

## 7.15 glldb::TableField Class Reference

Database table field class.

```
#include <tablefield.h>
```

Collaboration diagram for glldb::TableField:



## Public Member Functions

- [TableField](#) (const char \*data)  
*Constructor accepting `const char * data`.*
- [TableField](#) (const std::string &data)  
*Constructor accepting `std::string data`.*
- [~TableField](#) ()
- [size\\_t length](#) () const  
*Returns the length of the field.*
- [operator std::string](#) () const  
*Overridden conversion operator.*
- [TableField & operator=](#) (const char \*data)  
*Overridden assignment operator for `const char *`.*
- [TableField & operator=](#) (const std::string &data)  
*Overridden assignment operator for `std::string`.*
- [char & operator\[\]](#) (const size\_t idx)  
*Overridden index operator.*
- [const char & operator\[\]](#) (const size\_t idx) const  
*Overridden index operator.*
- [TableField & operator+=](#) (const char &c)  
*Overridden compound assignment operator.*
- [TableField & operator+=](#) (const std::string &data)  
*Overridden compound assignment operator.*

## Private Attributes

- `std::string` [m\\_data](#)

## Friends

- `std::ostream & operator<<` (std::ostream &out, const [TableField](#) &field)  
*Overridden << operator for printing a field.*

### 7.15.1 Detailed Description

Database table field class.

### 7.15.2 Constructor & Destructor Documentation

#### 7.15.2.1 `TableField::TableField ( const char * data )` `[explicit]`

Constructor accepting `const char * data`.

##### Parameters

<i>data</i>	The initial contents of the field.
-------------	------------------------------------

#### 7.15.2.2 `TableField::TableField ( const std::string & data )` `[explicit]`

Constructor accepting `std::string data`.

##### Parameters

<i>data</i>	The initial contents of the field.
-------------	------------------------------------

#### 7.15.2.3 `TableField::~~TableField ( )`

Destructor

### 7.15.3 Member Function Documentation

#### 7.15.3.1 `size_t TableField::length ( ) const`

Returns the length of the field.

##### Returns

The length of the field.

#### 7.15.3.2 `TableField::operator std::string ( ) const`

Overridden conversion operator.

Returns the field contents as a string.

#### 7.15.3.3 `TableField & TableField::operator+= ( const char & c )`

Overridden compound assignment operator.

##### Parameters

<i>c</i>	The character to append to the field.
----------	---------------------------------------

**Returns**

A reference to the same field.

**7.15.3.4 TableField & TableField::operator+= ( const std::string & *data* )**

Overridden compound assignment operator.

**Parameters**

<i>data</i>	The string to append to the field.
-------------	------------------------------------

**Returns**

A reference to the same field.

**7.15.3.5 TableField & TableField::operator= ( const char \* *data* )**

Overridden assignment operator for `const char *`.

**Parameters**

<i>data</i>	The new contents of the field.
-------------	--------------------------------

**Returns**

A reference to the same field.

**7.15.3.6 TableField & TableField::operator= ( const std::string & *data* )**

Overridden assignment operator for `std::string`.

**Parameters**

<i>data</i>	The new contents of the field.
-------------	--------------------------------

**Returns**

A reference to the same field.

**7.15.3.7 char & TableField::operator[] ( const size\_t *idx* )**

Overridden index operator.

**Parameters**

<i>idx</i>	The desired index.
------------	--------------------

**Returns**

A reference to the character at the specified index.

#### 7.15.3.8 `const char & TableField::operator[] ( const size_t idx ) const`

Overridden index operator.

##### Parameters

<i>idx</i>	The desired index.
------------	--------------------

##### Returns

A const reference to the character at the specified index.

### 7.15.4 Friends And Related Function Documentation

#### 7.15.4.1 `std::ostream& operator<< ( std::ostream & out, const TableField & field )` [*friend*]

Overridden << operator for printing a field.

##### Parameters

<i>out</i>	The ostream to which to print.
<i>field</i>	A reference to the field.

##### Returns

A reference to `out`.

### 7.15.5 Member Data Documentation

#### 7.15.5.1 `std::string glldb::TableField::m_data` [*private*]

The field contents

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

- [lib/database/tablefield.h](#)
- [lib/database/tablefield.cpp](#)

## 7.16 glldb::TableRow Class Reference

Database table row class.

```
#include <tablerow.h>
```

### Public Member Functions

- [TableRow](#) ()
- [TableRow](#) (const size\_t [size](#))  
*Constructor with initial number of fields.*
- [~TableRow](#) ()
- size\_t [size](#) () const  
*Returns the number of fields.*
- [TableField](#) & [operator\[\]](#) (const size\_t [idx](#))  
*Overridden index operator.*



- const [TableField](#) & [operator\[\]](#) (const size\_t idx) const  
*Overridden index operator.*
- void [append\\_field](#) (const char \*new\_field)  
*Appends a field to the row.*
- void [append\\_field](#) (const std::string &new\_field)  
*Appends a field to the row.*
- void [append\\_field](#) (const [TableField](#) &new\_field)  
*Appends a field to the row.*
- void [print](#) (std::ostream &stream) const  
*Prints a row.*

## Private Attributes

- std::vector< [TableField](#) > [m\\_fields](#)

### 7.16.1 Detailed Description

Database table row class.

### 7.16.2 Constructor & Destructor Documentation

#### 7.16.2.1 TableRow::TableRow ( )

Default constructor

#### 7.16.2.2 TableRow::TableRow ( const size\_t size ) [explicit]

Constructor with initial number of fields.

##### Parameters

<i>size</i>	The initial number of fields.
-------------	-------------------------------

#### 7.16.2.3 TableRow::~~TableRow ( )

Destructor

### 7.16.3 Member Function Documentation

#### 7.16.3.1 void TableRow::append\_field ( const char \* new\_field )

Appends a field to the row.

##### Parameters

<i>new_field</i>	The contents of the new field.
------------------	--------------------------------

#### 7.16.3.2 void TableRow::append\_field ( const std::string & new\_field )

Appends a field to the row.

## Parameters

<i>new_field</i>	The contents of the new field.
------------------	--------------------------------

**7.16.3.3 void TableRow::append\_field ( const TableField & *new\_field* )**

Appends a field to the row.

## Parameters

<i>new_field</i>	A field from which to copy.
------------------	-----------------------------

**7.16.3.4 TableField & TableRow::operator[] ( const size\_t *idx* )**

Overridden index operator.

## Parameters

<i>idx</i>	The zero-based index of the field.
------------	------------------------------------

## Returns

A reference to the field at the specified index.

**7.16.3.5 const TableField & TableRow::operator[] ( const size\_t *idx* ) const**

Overridden index operator.

## Parameters

<i>idx</i>	The zero-based index of the field.
------------	------------------------------------

## Returns

A const reference to the field at the specified index.

**7.16.3.6 void TableRow::print ( std::ostream & *stream* ) const**

Prints a row.

## Parameters

<i>stream</i>	The ostream to which to print.
---------------	--------------------------------

**7.16.3.7 size\_t TableRow::size ( ) const**

Returns the number of fields.

## Returns

The number of fields.

### 7.16.4 Member Data Documentation

#### 7.16.4.1 `std::vector<TableField> glldb::TableRow::m_fields` [private]

A vector of fields

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

- lib/database/[tablerow.h](#)
- lib/database/[tablerow.cpp](#)



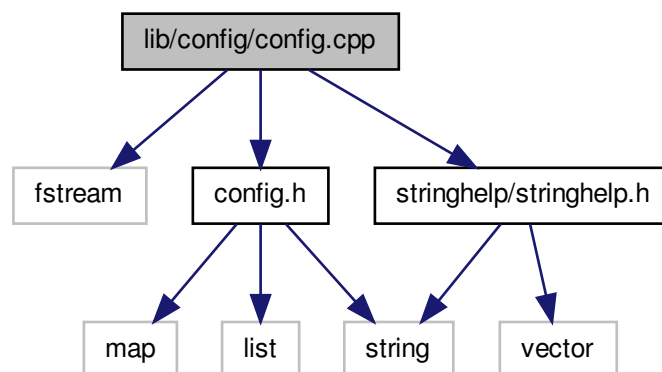
## Chapter 8

# File Documentation

### 8.1 lib/config/config.cpp File Reference

Implementation of program configurations class.

```
#include <fstream>
#include "config.h"
#include "stringhelp/stringhelp.h"
Include dependency graph for config.cpp:
```



#### 8.1.1 Detailed Description

Implementation of program configurations class.

##### Author

Paul Griffiths

##### Copyright

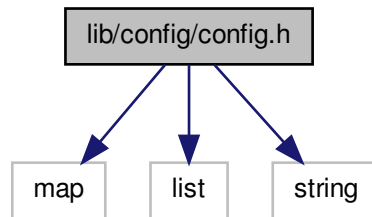
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.2 lib/config/config.h File Reference

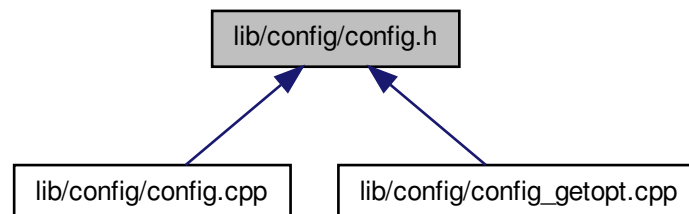
Interface to program configurations class.

```
#include <map>
#include <list>
#include <string>
```

Include dependency graph for config.h:



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



### Classes

- class [genleg::ConfigException](#)  
*Configuration module exception base class.*
- class [genleg::ConfigOptionNotSet](#)  
*Exception class for option not set.*
- class [genleg::ConfigBadOption](#)  
*Exception class for bad provided option.*
- class [genleg::ConfigCouldNotOpenFile](#)  
*Exception class for when conf file cannot be opened.*
- class [genleg::ConfigBadConfigFile](#)  
*Exception class for badly formed configuration file.*
- class [genleg::Config](#)  
*Configuration options class.*

## Enumerations

- enum [genleg::Argument](#)  
*Enumeration class for option argument specifications.*

### 8.2.1 Detailed Description

Interface to program configurations class.

#### Author

Paul Griffiths

#### Copyright

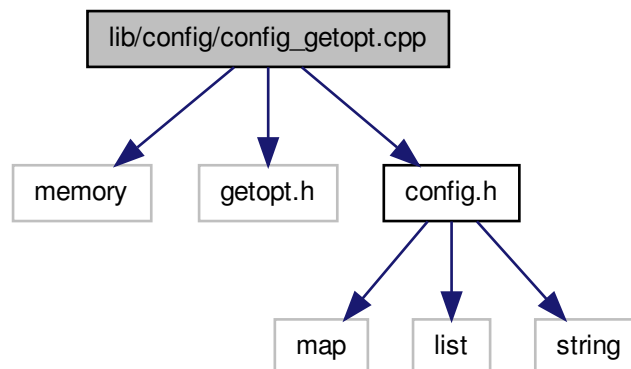
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.3 lib/config/config\_getopt.cpp File Reference

Implementation of command line functionality.

```
#include <memory>
#include <getopt.h>
#include "config.h"
```

Include dependency graph for config\_getopt.cpp:



## Macros

- `#define _XOPEN_SOURCE 600`

### 8.3.1 Detailed Description

Implementation of command line functionality. Included in separate file to isolate usage of non-standard getopt library.

**Author**

Paul Griffiths

**Copyright**

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

**8.3.2 Macro Definition Documentation****8.3.2.1 #define XOPEN\_SOURCE 600**

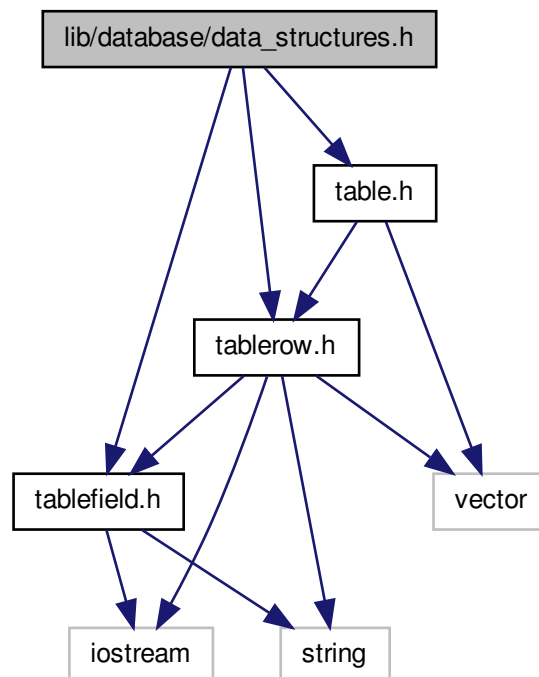
UNIX feature test macro for getopt library

**8.4 lib/database/data\_structures.h File Reference**

Main interface to database data structures.

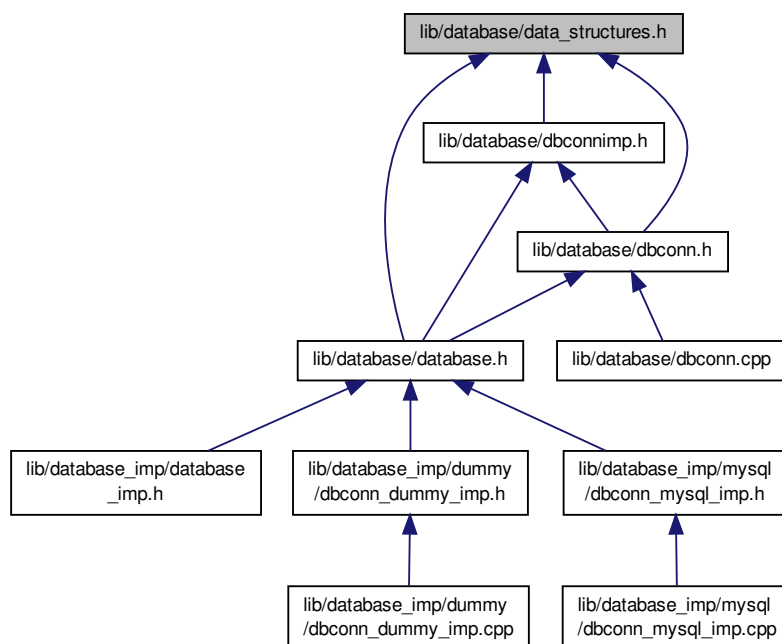
```
#include "tablefield.h"  
#include "tablerow.h"  
#include "table.h"
```

Include dependency graph for data\_structures.h:





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



### 8.4.1 Detailed Description

Main interface to database data structures.

#### Author

Paul Griffiths

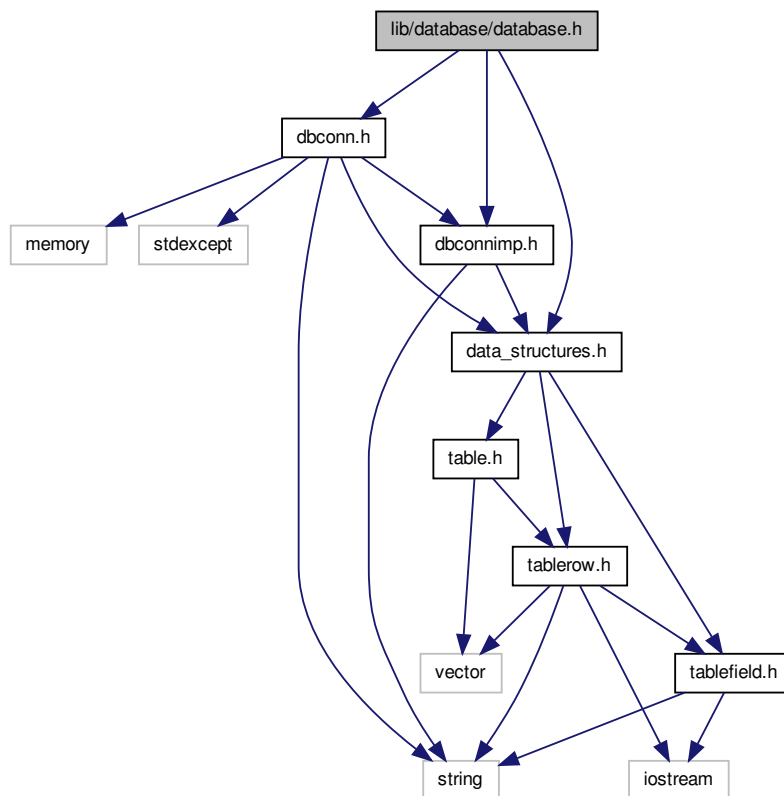
## Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

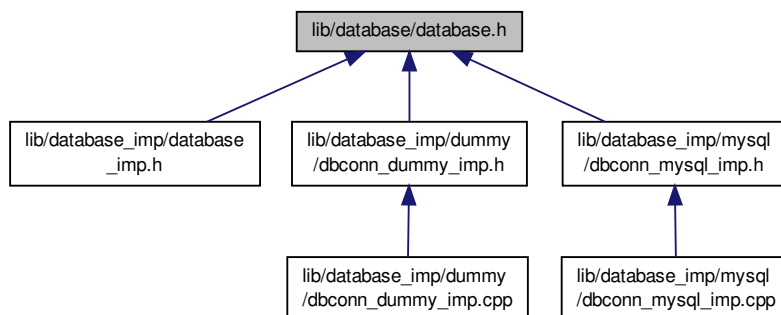
## 8.5 lib/database/database.h File Reference

User interface to database functionality.

```
#include "data_structures.h"  
#include "dbconnimp.h"  
#include "dbconn.h"  
Include dependency graph for database.h:
```



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



### 8.5.1 Detailed Description

User interface to database functionality.

#### Author

Paul Griffiths

#### Copyright

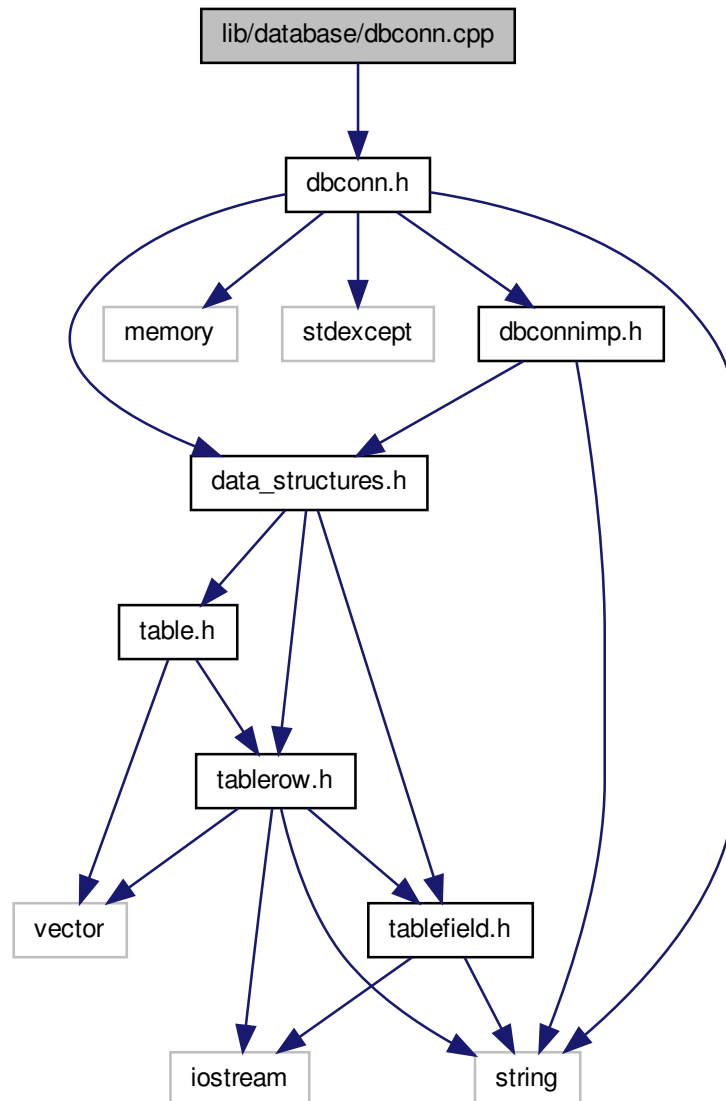
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.6 lib/database/dbconn.cpp File Reference

Implementation of database connection class.

```
#include "dbconn.h"
```

Include dependency graph for dbconn.cpp:



### 8.6.1 Detailed Description

Implementation of database connection class.

Author

Paul Griffiths

Copyright

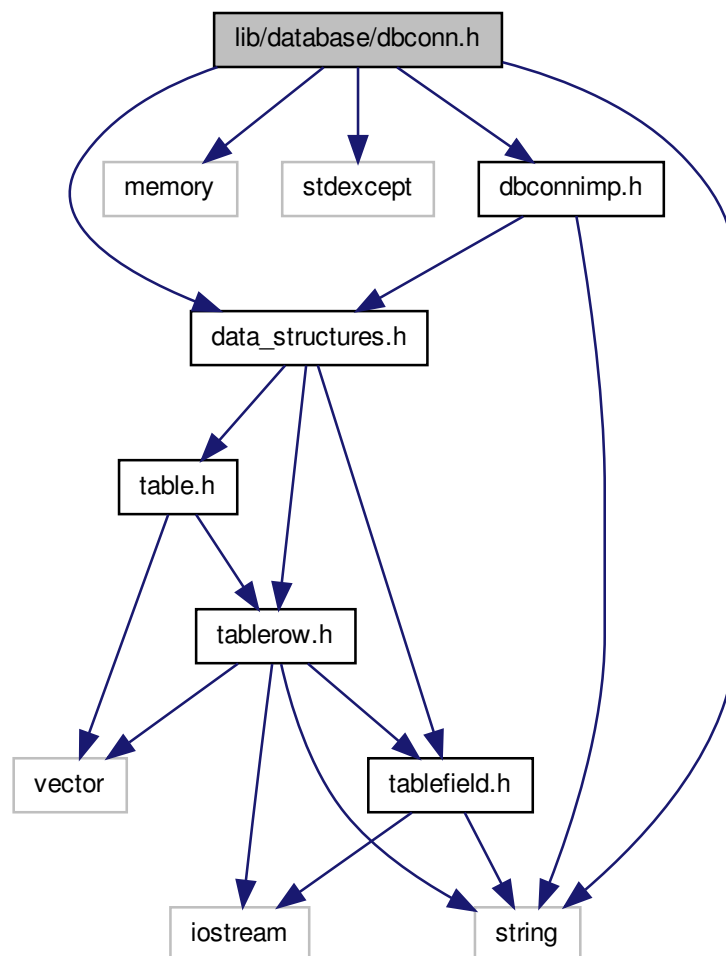
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.7 lib/database/dbconn.h File Reference

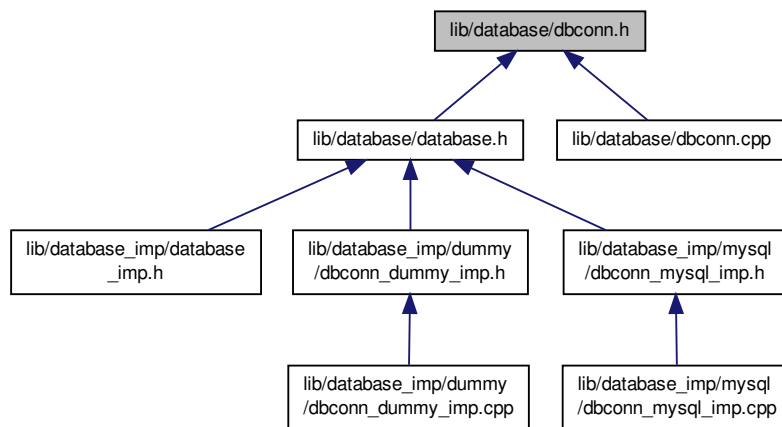
Interface to database connection base class.

```
#include <string>
#include <memory>
#include <stdexcept>
#include "data_structures.h"
#include "dbconnimp.h"
```

Include dependency graph for dbconn.h:



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



## Classes

- class `gldb::DBConnException`  
*Base database connection exception class.*
- class `gldb::DBConnCouldNotConnect`  
*Could not connect to database exception class.*
- class `gldb::DBConnCouldNotQuery`  
*Could not execute database query exception class.*
- class `gldb::DBConn`  
*Database connection class.*

### 8.7.1 Detailed Description

Interface to database connection base class.

#### Author

Paul Griffiths

#### Copyright

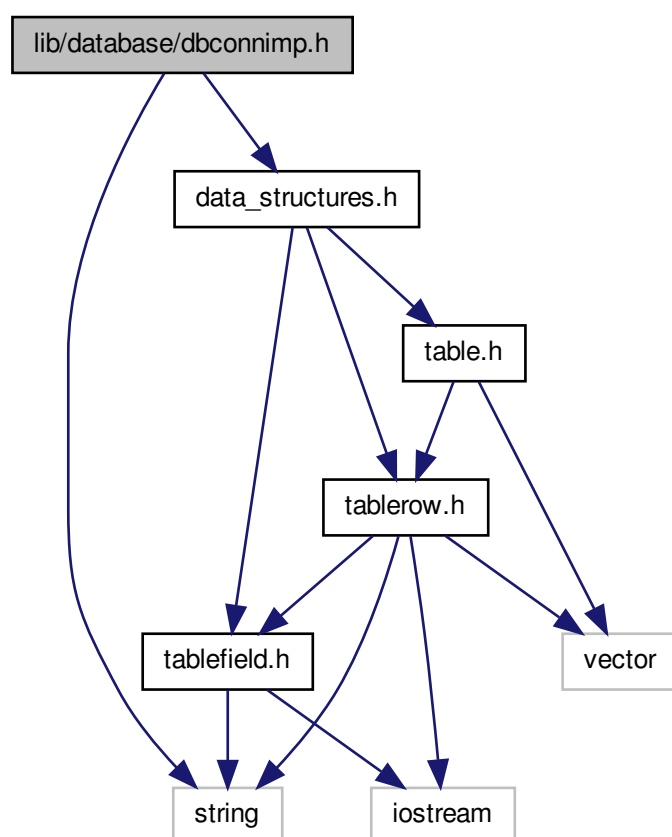
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.8 lib/database/dbconnimp.h File Reference

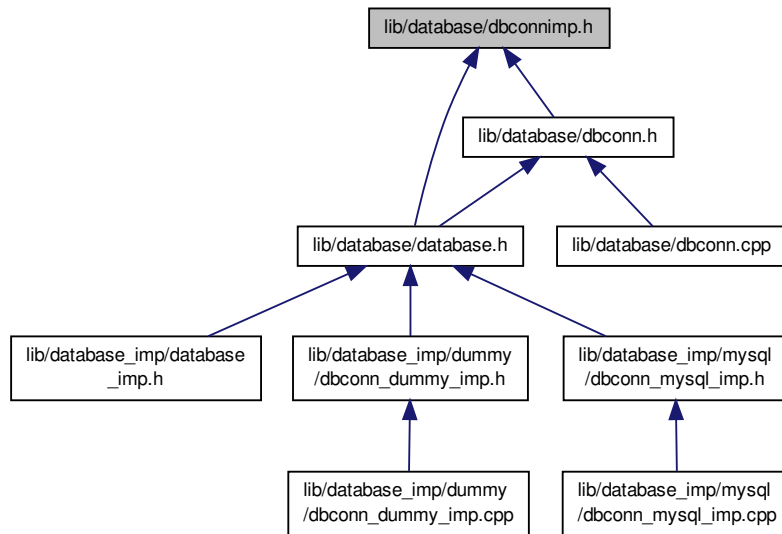
Interface to abstract database implementation base class.

```
#include <string>
#include "data_structures.h"
```

Include dependency graph for dbconnimp.h:



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



## Classes

- class [gldb::DBConnImp](#)

*Abstract database implementation base class.*

### 8.8.1 Detailed Description

Interface to abstract database implementation base class.

#### Author

Paul Griffiths



### Copyright

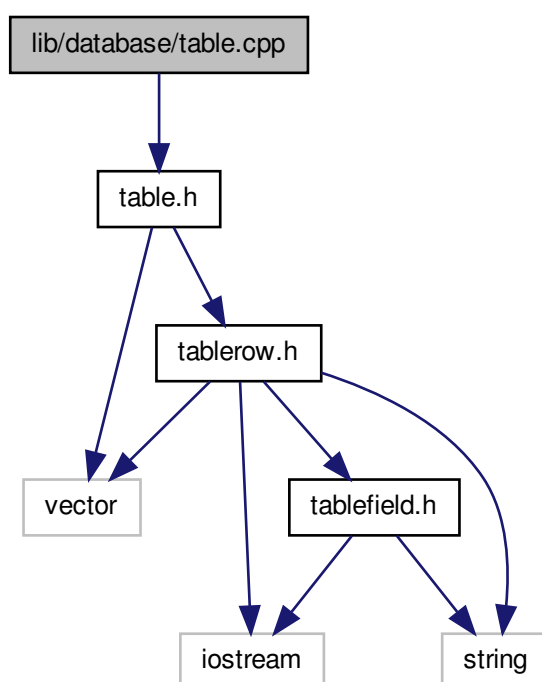
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.9 lib/database/table.cpp File Reference

Implementation of database table data structure.

```
#include "table.h"
```

Include dependency graph for table.cpp:



### 8.9.1 Detailed Description

Implementation of database table data structure.

#### Author

Paul Griffiths

### Copyright

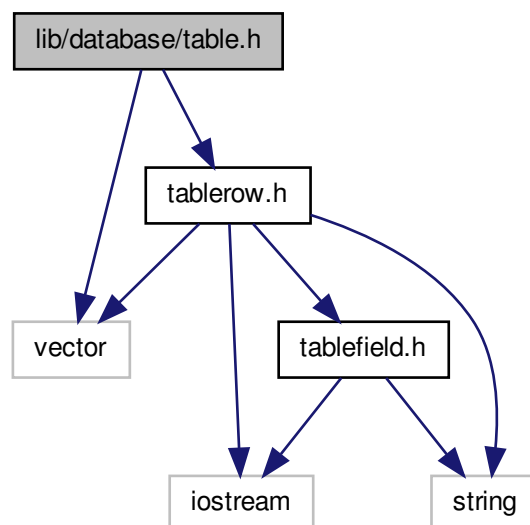
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.10 lib/database/table.h File Reference

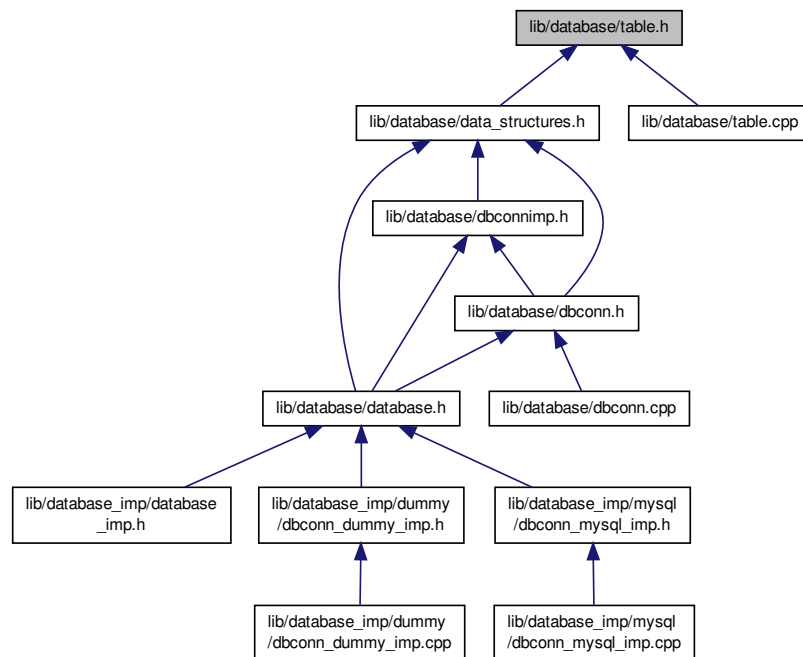
Interface to database table data structure.

```
#include <vector>
#include "tablerow.h"
```

Include dependency graph for table.h:



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



## Classes

- class [gldb::Table](#)

*Database table class.*

### 8.10.1 Detailed Description

Interface to database table data structure.

#### Author

Paul Griffiths

#### Copyright

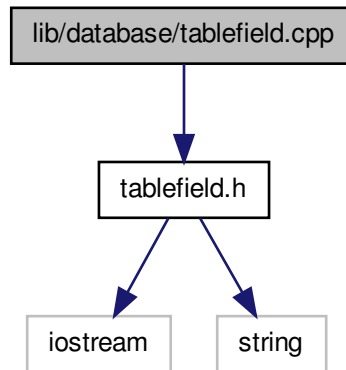
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.11 lib/database/tablefield.cpp File Reference

Implementation of database table field class.

```
#include "tablefield.h"
```

Include dependency graph for tablefield.cpp:



### 8.11.1 Detailed Description

Implementation of database table field class.

#### Author

Paul Griffiths

#### Copyright

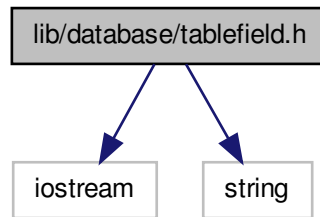
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.12 lib/database/tablefield.h File Reference

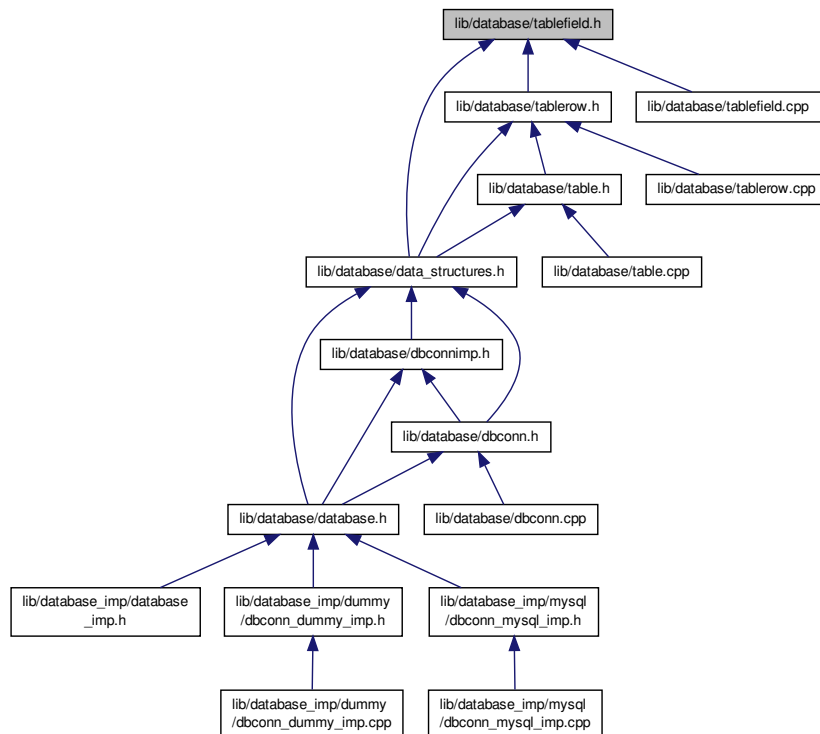
Interface to database table field class.

```
#include <iostream>
#include <string>
```

Include dependency graph for tablefield.h:



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



## Classes

- class `gldb::TableField`  
*Database table field class.*

## Functions

- `std::ostream & gldb::operator<< (std::ostream &out, const TableField &field)`

Overridden << operator for printing a field.

### 8.12.1 Detailed Description

Interface to database table field class.

#### Author

Paul Griffiths

#### Copyright

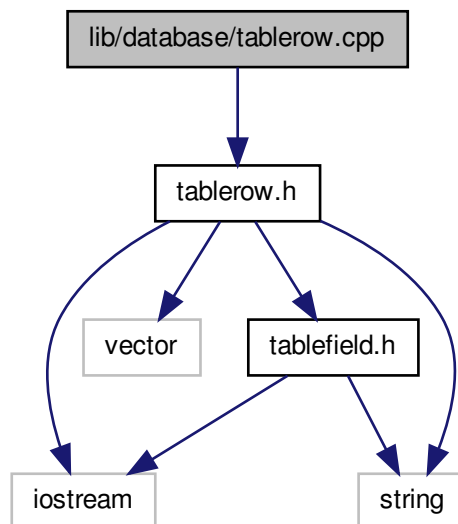
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.13 lib/database/tablerow.cpp File Reference

Implementation of database table row data structure.

```
#include "tablerow.h"
```

Include dependency graph for tablerow.cpp:



### 8.13.1 Detailed Description

Implementation of database table row data structure.

#### Author

Paul Griffiths

### Copyright

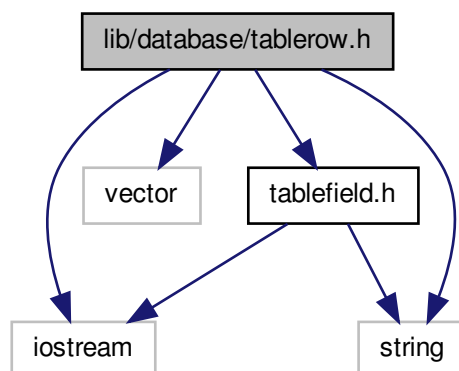
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.14 lib/database/tablerow.h File Reference

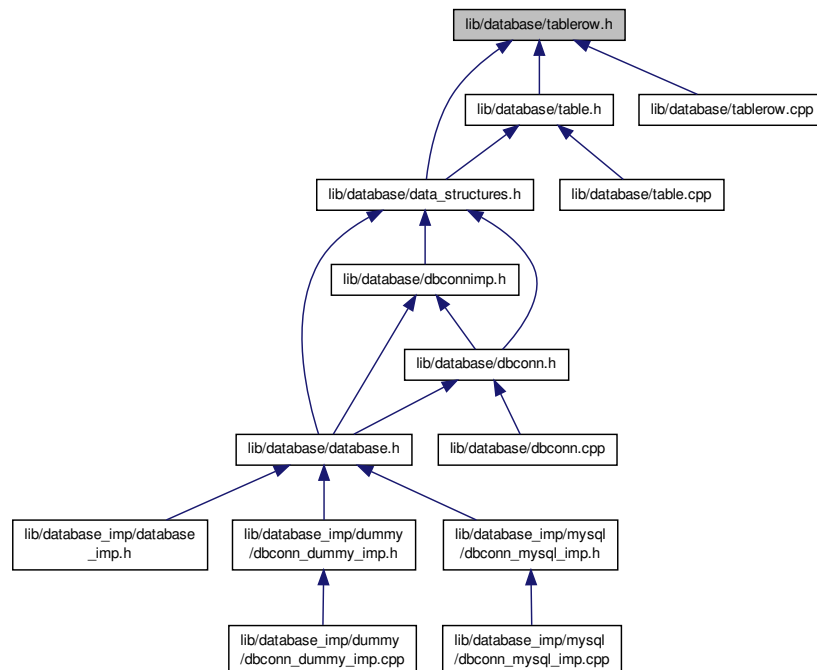
Interface to database table row data structure.

```
#include <iostream>
#include <vector>
#include <string>
#include "tablefield.h"
```

Include dependency graph for tablerow.h:



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



## Classes

- class `gldb::TableRow`

*Database table row class.*

### 8.14.1 Detailed Description

Interface to database table row data structure.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

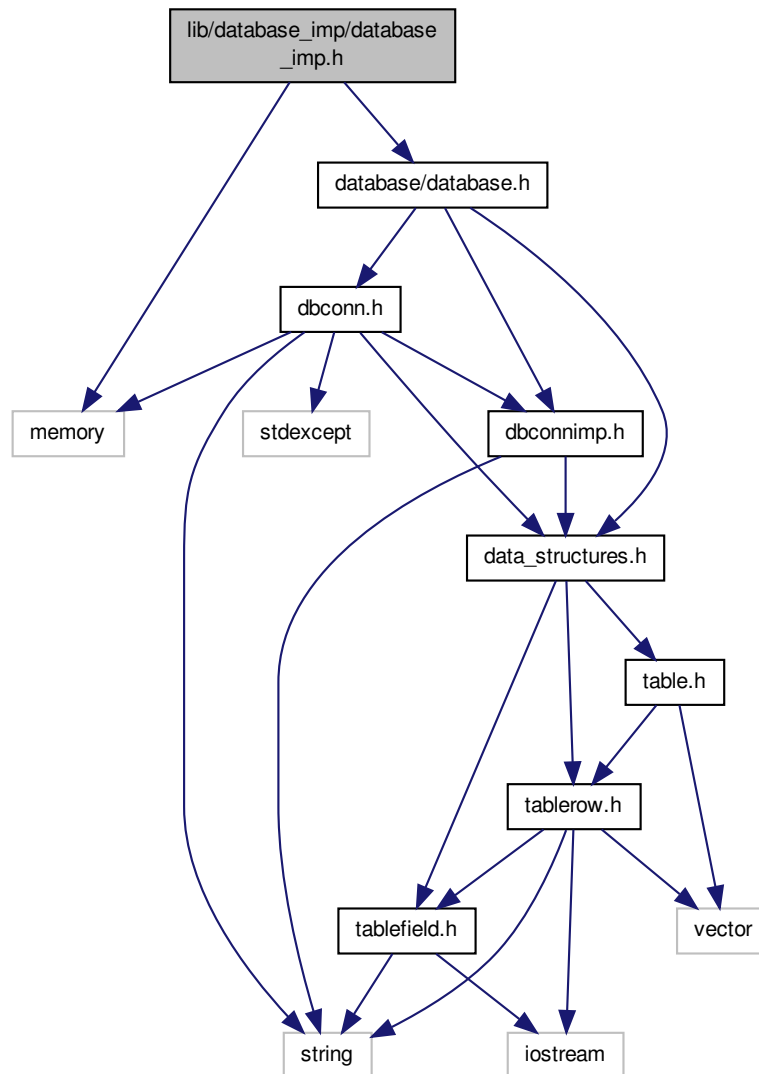
## 8.15 lib/database\_imp/database\_imp.h File Reference

Interface to database implementation factory function.

```
#include <memory>
#include "database/database.h"
```



Include dependency graph for database\_imp.h:



## Functions

- `DBConnImp * glldb::get_connection` (const std::string database, const std::string hostname, const std::string username, const std::string password)  
*Creates and returns a pointer to a database implementation.*
- std::string `glldb::get_database_type` ()  
*Returns the name of the compiled-in database type.*

### 8.15.1 Detailed Description

Interface to database implementation factory function.

**Author**

Paul Griffiths

**Copyright**

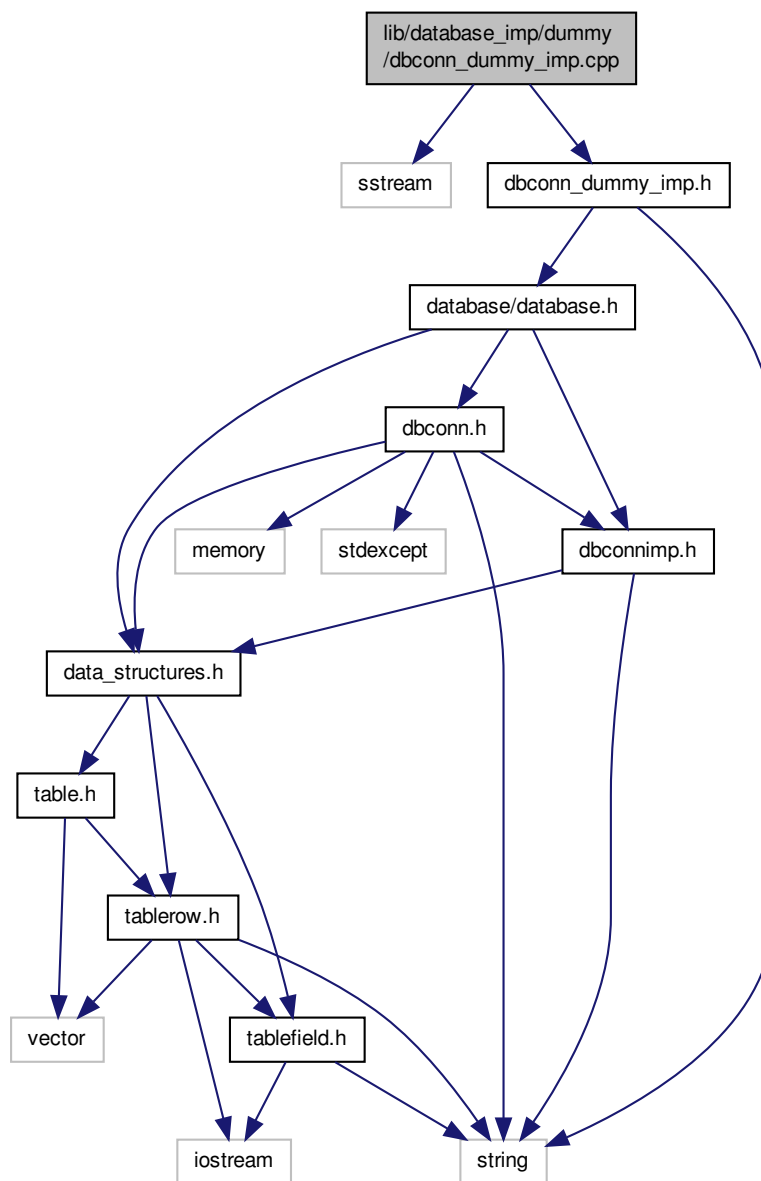
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.16 lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp File Reference

Implementation of Dummy database connection implementation class.

```
#include <sstream>
#include "dbconn_dummy_imp.h"
```

Include dependency graph for dbconn\_dummy\_imp.cpp:



### 8.16.1 Detailed Description

Implementation of Dummy database connection implementation class.

Author

Paul Griffiths

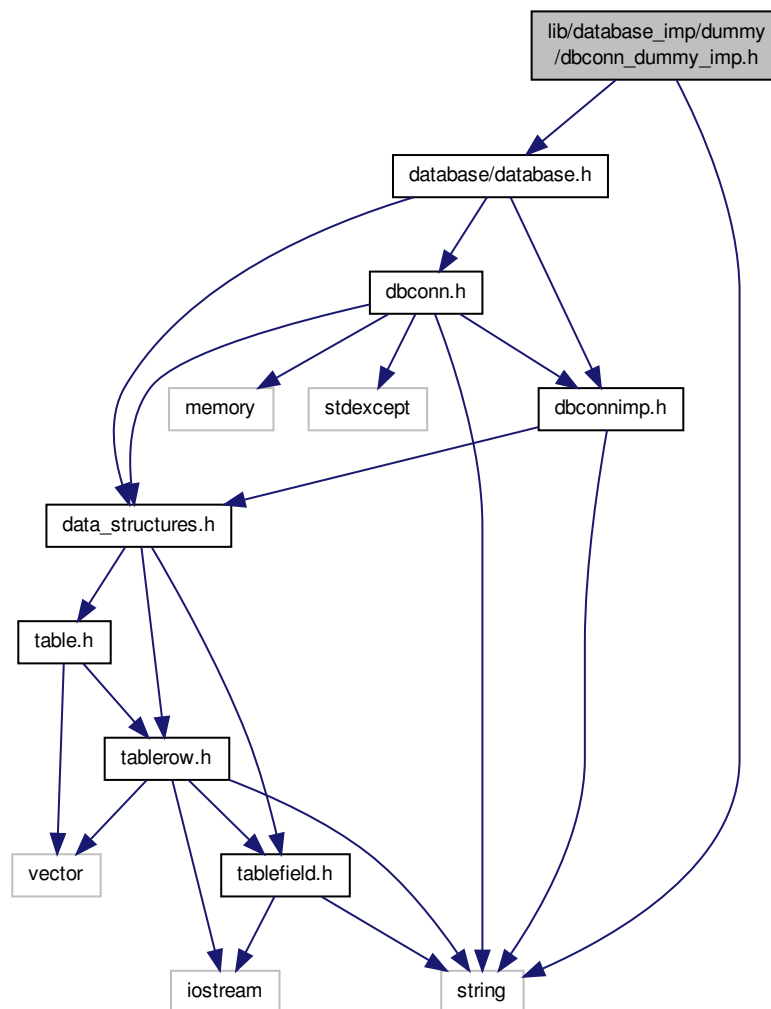
Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

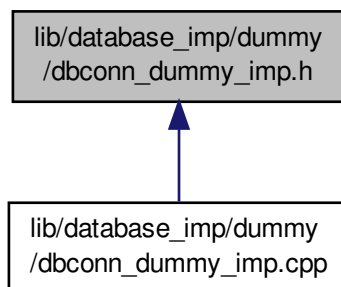
## 8.17 lib/database\_imp/dummy/dbconn\_dummy\_imp.h File Reference

Interface to dummy database connection implementation class.

```
#include <string>
#include "database/database.h"
Include dependency graph for dbconn_dummy_imp.h:
```



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



## Classes

- class [gldb::DBConnDummy](#)

*Dummy database implementation class.*

### 8.17.1 Detailed Description

Interface to dummy database connection implementation class.

#### Author

Paul Griffiths

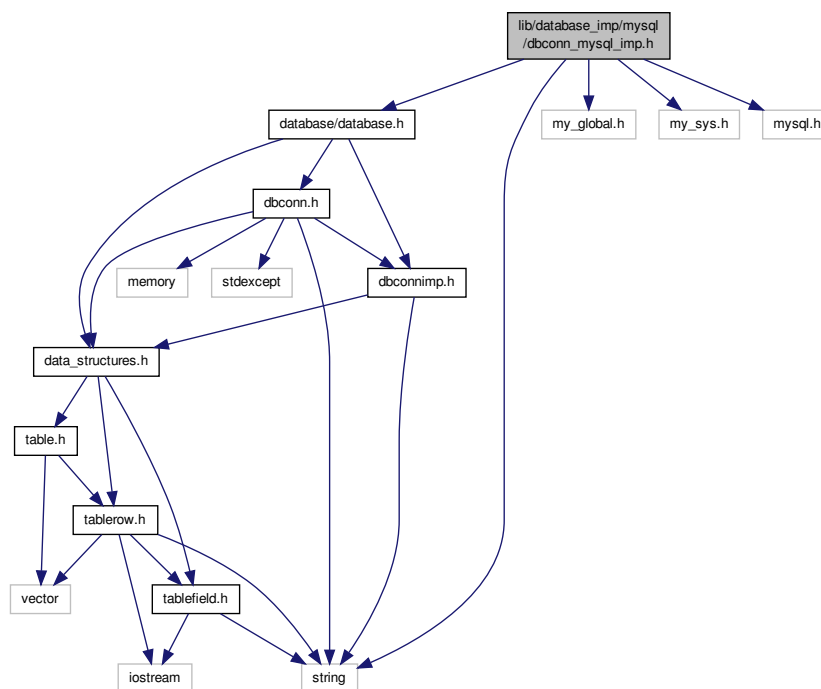


## 8.19 lib/database\_imp/mysql/dbconn\_mysql\_imp.h File Reference

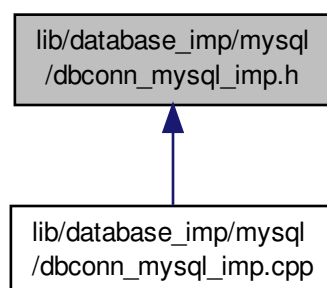
Interface to MySQL database connection implementation class.

```
#include <string>
#include "database/database.h"
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
```

Include dependency graph for dbconn\_mysql\_imp.h:



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



## Classes

- class `gldb::DBConnMySQL`  
*MySQL database implementation class.*

### 8.19.1 Detailed Description

Interface to MySQL database connection implementation class.

#### Author

Paul Griffiths

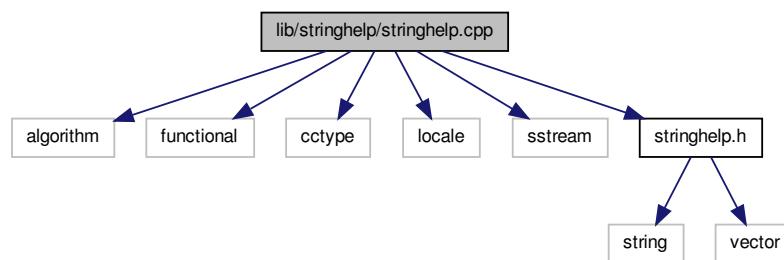
#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

## 8.20 lib/stringhelp/stringhelp.cpp File Reference

Implementation of string helper functions.

```
#include <algorithm>
#include <functional>
#include <cctype>
#include <locale>
#include <sstream>
#include "stringhelp.h"
Include dependency graph for stringhelp.cpp:
```



### 8.20.1 Detailed Description

Implementation of string helper functions.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>



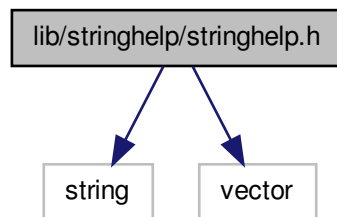
## 8.21 lib/stringhelp/stringhelp.h File Reference

Interface to string helper functions.

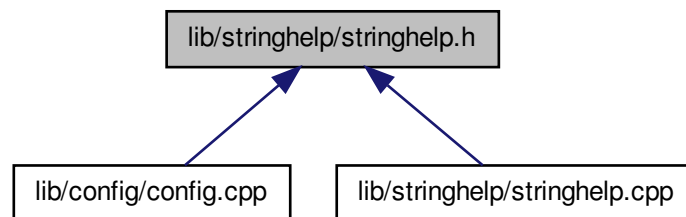
```
#include <string>
```

```
#include <vector>
```

Include dependency graph for stringhelp.h:



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



### Functions

- `std::string & pgstring::trim_front (std::string &s)`  
*Trims leading whitespace from a string.*
- `std::string & pgstring::trim_back (std::string &s)`  
*Trims trailing whitespace from a string.*
- `std::string & pgstring::trim (std::string &s)`  
*Trims leading and trailing whitespace from a string.*
- `std::vector< std::string > pgstring::split (const std::string &s, const char delim)`  
*Splits a delimited string into tokens.*

#### 8.21.1 Detailed Description

Interface to string helper functions.

**Author**

Paul Griffiths

**Copyright**

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. <http://www.gnu.org/licenses/>

# Index

- ~Config
  - genleg::Config, [17](#)
- ~DBConnDummy
  - gldb::DBConnDummy, [29](#)
- ~DBConnImp
  - gldb::DBConnImp, [31](#)
- ~DBConnMySQL
  - gldb::DBConnMySQL, [33](#)
- ~Table
  - gldb::Table, [35](#)
- ~TableField
  - gldb::TableField, [38](#)
- ~TableRow
  - gldb::TableRow, [41](#)
- \_XOPEN\_SOURCE
  - config\_getopt.cpp, [48](#)
- add\_cmdline\_option
  - genleg::Config, [18](#)
- append\_field
  - gldb::TableRow, [41](#), [42](#)
- append\_record
  - gldb::Table, [35](#)
- Config
  - genleg::Config, [17](#)
- config\_getopt.cpp
  - \_XOPEN\_SOURCE, [48](#)
- DBConn
  - gldb::DBConn, [25](#)
- DBConnCouldNotConnect
  - gldb::DBConnCouldNotConnect, [26](#)
- DBConnCouldNotQuery
  - gldb::DBConnCouldNotQuery, [28](#)
- DBConnDummy
  - gldb::DBConnDummy, [29](#)
- DBConnException
  - gldb::DBConnException, [30](#)
- DBConnImp
  - gldb::DBConnImp, [31](#)
- DBConnMySQL
  - gldb::DBConnMySQL, [33](#)
- Database interaction module, [12](#)
  - get\_connection, [12](#)
  - get\_database\_type, [12](#)
- General purpose helpers., [14](#)
  - split, [14](#)
  - trim, [14](#)
  - trim\_back, [14](#)
  - trim\_front, [15](#)
- genleg::Config, [17](#)
  - ~Config, [17](#)
  - add\_cmdline\_option, [18](#)
  - Config, [17](#)
  - is\_set, [18](#)
  - m\_opts\_set, [19](#)
  - m\_opts\_supp, [19](#)
  - populate\_from\_cmdline, [18](#)
  - populate\_from\_file, [19](#)
- genleg::ConfigBadConfigFile, [19](#)
- genleg::ConfigBadOption, [20](#)
- genleg::ConfigCouldNotOpenFile, [21](#)
- genleg::ConfigException, [22](#)
- genleg::ConfigOptionNotSet, [23](#)
- get\_connection
  - Database interaction module, [12](#)
- get\_database\_type
  - Database interaction module, [12](#)
- get\_headers
  - gldb::Table, [35](#)
- gldb::DBConn, [24](#)
  - DBConn, [25](#)
  - m\_imp, [25](#)
  - operator=, [25](#)
  - select, [25](#)
- gldb::DBConnCouldNotConnect, [26](#)
  - DBConnCouldNotConnect, [26](#)
- gldb::DBConnCouldNotQuery, [27](#)
  - DBConnCouldNotQuery, [28](#)
- gldb::DBConnDummy, [28](#)
  - ~DBConnDummy, [29](#)
  - DBConnDummy, [29](#)
  - operator=, [29](#)
  - select, [29](#)
- gldb::DBConnException, [30](#)
  - DBConnException, [30](#)
- gldb::DBConnImp, [31](#)
  - ~DBConnImp, [31](#)
  - DBConnImp, [31](#)
  - select, [31](#)
- gldb::DBConnMySQL, [32](#)
  - ~DBConnMySQL, [33](#)
  - DBConnMySQL, [33](#)
  - m\_conn, [34](#)
  - operator=, [33](#)
  - select, [33](#)
- gldb::Table, [34](#)

- ~Table, 35
  - append\_record, 35
  - get\_headers, 35
  - m\_headers, 36
  - m\_records, 36
  - num\_fields, 35
  - num\_records, 36
  - Table, 35
- gldb::TableField, 36
  - ~TableField, 38
  - length, 38
  - m\_data, 40
  - operator std::string, 38
  - operator<<, 40
  - operator+=, 38, 39
  - operator=, 39
  - TableField, 38
- gldb::TableRow, 40
  - ~TableRow, 41
  - append\_field, 41, 42
  - m\_fields, 43
  - print, 42
  - size, 42
  - TableRow, 41
- is\_set
  - genleg::Config, 18
- length
  - gldb::TableField, 38
- lib/config/config.cpp, 45
- lib/config/config.h, 46
- lib/config/config\_getopt.cpp, 47
- lib/database/data\_structures.h, 48
- lib/database/database.h, 50
- lib/database/dbconn.cpp, 51
- lib/database/dbconn.h, 53
- lib/database/dbconnimp.h, 54
- lib/database/table.cpp, 57
- lib/database/table.h, 58
- lib/database/tablefield.cpp, 59
- lib/database/tablefield.h, 60
- lib/database/ablerow.cpp, 62
- lib/database/ablerow.h, 63
- lib/database\_imp/database\_imp.h, 64
- lib/database\_imp/dummy/dbconn\_dummy\_imp.cpp, 66
- lib/database\_imp/dummy/dbconn\_dummy\_imp.h, 68
- lib/database\_imp/mysql/dbconn\_mysql\_imp.cpp, 70
- lib/database\_imp/mysql/dbconn\_mysql\_imp.h, 71
- lib/stringhelp/stringhelp.cpp, 72
- lib/stringhelp/stringhelp.h, 73
- m\_conn
  - gldb::DBConnMySQL, 34
- m\_data
  - gldb::TableField, 40
- m\_fields
  - gldb::TableRow, 43
- m\_headers
  - gldb::Table, 36
- m\_imp
  - gldb::DBConn, 25
- m\_opts\_set
  - genleg::Config, 19
- m\_opts\_supp
  - genleg::Config, 19
- m\_records
  - gldb::Table, 36
- num\_fields
  - gldb::Table, 35
- num\_records
  - gldb::Table, 36
- operator std::string
  - gldb::TableField, 38
- operator<<
  - gldb::TableField, 40
- operator+=
  - gldb::TableField, 38, 39
- operator=
  - gldb::DBConn, 25
  - gldb::DBConnDummy, 29
  - gldb::DBConnMySQL, 33
  - gldb::TableField, 39
- populate\_from\_cmdline
  - genleg::Config, 18
- populate\_from\_file
  - genleg::Config, 19
- print
  - gldb::TableRow, 42
- Program configuration module, 13
- select
  - gldb::DBConn, 25
  - gldb::DBConnDummy, 29
  - gldb::DBConnImp, 31
  - gldb::DBConnMySQL, 33
- size
  - gldb::TableRow, 42
- split
  - General purpose helpers., 14
- Table
  - gldb::Table, 35
- TableField
  - gldb::TableField, 38
- TableRow
  - gldb::TableRow, 41
- trim
  - General purpose helpers., 14
- trim\_back
  - General purpose helpers., 14
- trim\_front
  - General purpose helpers., 15