Server-side function

DbConnect

This function opens a connection to a database and returns an element that can be used with dbQuery. To disconnect use destroyElement.

> **Note:** Connecting and disconnecting many times can have a performance impact on the server. For optimal performance it is recommended that you use dbConnect only once when the resource starts, and share the connection element with the whole script.

> **Note:** If you use MySQL 8 or newer, add this line to mysqld.cnf
> **default-authentication-plugin=mysql_native_password**

Syntax

```
element dbConnect ( string databaseType, string host [, string username = "", string password = "", string options = "" ] )
```

OOP Syntax Help! I don't understand this!

Method: *Connection(...)*

Required Arguments

- **databaseType:** The type of database. This can be either *sqlite* or *mysql*
- **host:** The target to connect to. The format of this depends on the database type.
  - For SQLite it is a filepath to a SQLite database file. If the filepath starts with ":/" then the server's global databases directory is used. The file will be created if it does not exist.
  - For MySQL it is a list of key=value pairs separated by semicolons. Supported keys are:
    - **dbname**: Name of the database to use e.g. *dbname=test*
    - **host**: Host address e.g. *host=127.0.0.1*
    - **port**: Host port e.g. *port=1234* (optional, defaults to standard MySQL port if not used)
    - **unix_socket**: Unix socket or named pipe to use (optional)
    - **charset**: Communicate with the server using a character which is different from the default e.g. *charset=utf8* (optional)

Optional Arguments

- **username:** Usually required for MySQL, ignored by SQLite
- **password:** Usually required for MySQL, ignored by SQLite
- **options :** List of key=value pairs separated by semicolons. Supported keys are:
  - **share** which can be set to 0 or 1. (Default value for SQLite is "share=1", for MySQL is "share=0"). When set to 1, the connection is shared and will be used by other calls to dbConnect with the same host string. This is usually a good thing for SQLite connections, but not so good for MySQL unless care is taken.
  - **batch** which can be set to 0 or 1. (Default is "batch=1"). When set to 1, queries called in the same frame are automatically batched together which can significantly speed up inserts/updates. The downside is you lose control of the feature that is used to achieve batching (For SQLite it is transactions, for MySQL it is autocommit mode). Therefore, if you use transactions, lock tables or control autocommit yourself, you may want to disable this feature.
  - **autoreconnect** which can be set to 0 or 1. (Default value "autoreconnect=1"). When set to 1, dropped connections will automatically be reconnected. Note that session variables (incl. SET NAMES), user variables, table locks and temporary tables will be reset because of the reconnection. So if you use these fancy features, you will need to turn autoreconnect off and cope with dropped connections some other way.
  - **log** which can be set to 0 or 1. (Default value "log=1"). When set to 0, activity from this connection will not be recorded in the database debug log file.
  - **tag** (Default value "tag=script"). A string which helps identify activity from this connection in the database debug log file.
  - **suppress** A comma separated list of error codes to ignore. (eg. "suppress=1062,1169").
  - **multi_statements** Enable multiple statements (separated by a semi-colon) in one query. Use dbPrepareString when building a multiple statement query to reduce SQL injection risks.
  - **queue** Name of the queue to use. (Default value for SQLite is "sqlite", for MySQL default is the host string from the **host** argument). Asynchronous database queries in the same queue are processed in order, one at a time. Any name can be used.
  - **use_ssl** which can be set to 0 or 1. (Default value is 0), ignored by SQLite

Returns

Returns a database connection element unless there are problems, in which case it return *false*.

Remarks

Under certain platforms, for example on Unix-based OSes like Linux, using this function could fail with a debug warning containing "[Could not connect]" accompanied by a prior debug error explaining the problem. In that case you should check the Server Manual to see if you have missed any recommended (best-effort) steps for server set-up.