# TMS myCloudData SDK
## DEVELOPERS GUIDE

February 2017


Web: http://www.tmssoftware.com
Email: info@tmssoftware.com

# Index

# Availability

TMS myCloudData SDK v1.0 is available for Windows and Linux node.js based web servers. It supports MS SQL or MySQL database support on Windows and MySQL support on Linux. TMS myCloudData SDK offers seamless REST enabled access to structured cloud data storage. With the SDK user based CRUD permission settings are possible as well as user based table sharing. Rich meta data can be used for enhanced and automatic client data entry forms. As the data can be accessed via a REST API, access from various operating systems using different programming languages is possible. From TMS, libraries are offered for access from .NET code, Delphi, C++Builder, Pascal or Javascript.

# Requirements

Software

- node.js server v6.9.1 or newer
- MS SQL 2014 or newer
- MySQL v5.6 or newer

TMS myCloudData SDK uses following node.js packages:

- config-js
- hapi
- inert
- mysql (For use with MySQL only)
- tedious (For use with MS SQL only)
- tedious-connection-pool (For use with MS SQL only)
- request  (For use with Google reCaptcha only)

Certificates
For use via HTTPS, a certificate needs to be obtained.

# Online references

TMS software website:
http://www.tmssoftware.com

TMS myCloudData SDK product page:
https://www.tmssoftware.com/site/myclouddatasdk.asp

TMS myCloudData libraries:
http://www.myclouddata.net/#/documentation/libraries

# Terms of use

With the purchase of TMS myCloudData SDK, you are entitled to deploy the SDK on your server for use from one domain. There is no limit on the number of users that can access the structured cloud data. With the purchase of the TMS myCloudData SDK comes also our support services and free SDK updates during the period of one year after purchase date.

It is not permitted to:

1) use the SDK on multiple domains. Per domain a license is required.

2) resell / sell accounts on the myCloudData SDK based service.

3) make the source code of the SDK publicly available

4) use of the myCloudData name when offering services to 3rd parties

5) use TMS CloudData SDK on multiple domains with one purchased license

# License

LICENSOR: tmssoftware.com bvba
LICENSEE: company or person who purchased the license to TMS myCloudData SDK

By using the TMS myCloudData SDK you are agreeing to be bound by the terms of this Agreement. If you do not agree then please uninstall the TMS myCloudData SDK.

1. GRANT OF LICENCE. In consideration of your agreement to abide by the terms and conditions of this licence the LICENSOR grants to you the LICENSEE the non-exclusive right to use the TMS myCloudData SDK on one domain.

2. OWNERSHIP OF TMS myCloudData SDK. An express condition of this licence is that the LICENSOR retains title and ownership of the TMS myCloudData SDK and all copyrights.

3. COPYING OF TMS myCloudData SDK. You may make unlimited copies of the TMS myCloudData SDK for your personal use.

4. USE RESTRICTIONS. You may not modify, adapt, translate, reverse engineer, decompile, disassemble, or create derivative works based on the TMS myCloudData SDK.

5. DISTRIBUTION RESTRICTIONS. You may not distribute copies of the myCloudData SDK to others.

6. TERMINATION. This licence remains effective until terminated. This licence will terminate automatically without notice from the LICENSOR if you fail to comply with any provision of this Licence. Upon termination you shall destroy all copies of the TMS myCloudData SDK.

# Limited warranty

a) Except as specifically stated in this agreement, the TMS myCloudData SDK is provided and licensed 'AS IS' without warranty of any kind, either express or implied, including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

b) The LICENSOR does not warrant that the functions contained in the TMS myCloudData SDK will meet your requirements or that operation of the TMS myCloudData SDK will be either error free or appear precisely as described in the documentation.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES.

To the maximum legal extent by applicable law, the LICENSOR shall not be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use this TMS myCloudData SDK, even if the LICENSOR has been advised of the possibility of such damages.

# Main features

The TMS myCloudData SDK enables REST API (HTTP or HTTPS) based access to structured cloud data storage.

- Can be deployed on a node.js based web server
- Supports a MS SQL or MySQL database
- Can be used on Microsoft Windows or Linux operating system
- An unlimited number of data access users can be defined
- Allows to use an unlimited number of tables, fields
- Per user CRUD based permissions can be defined for tables
- Rich meta data can be configured per table field
- Permission based sharing of data between users can be configured
- Accessible via REST API from any programming environment supporting REST
- Structured table based data storage
- Tables can have following field types: string, int, float, boolean, date, time, datetime, blob
- Via metadata, typed fields can be defined + field lookup relationships

# myCloudData install procedure

**Install and configure a database server**

Both MySQL server and Microsoft SQL server are supported.

**MySQL:**

- Install MySQL Server
    - Linux minimum required version: v5.6
    - Windows minimum required version: v5.7
- Create a database schema called "restdb"
- Import the file "MYSQL_RESTDB.sql" to create all required tables and fields.
- Create a user called "RESTDB" and enter a password

**MS SQL:**

- Install SQL Server
    - Minimum required version: SQL Server 2014
- Make sure the SQL Server is configured to accept SQL Server Authentication:
    - In the SQL Server Management Studio, right click the SQL Server and select Properties
    - Select the Security tab
    - Make sure “SQL Server and Windows Authentication mode” is active
- Create a database schema called "restdb"
- Execute the file "MSSQL_RESTDB.sql" to create all required tables and fields.
- Create a user called "RESTDB" and enter a password. Make sure the user has access to the “restdb” database.

**Install NodeJS and NPM**

NodeJS minimum required version: v6.9.1

**Install NodeJS modules**

Navigate to the folder where the “server.js” file is located.
The required NodeJS modules and required version number:

- config-js@1.1.9
- hapi@9.3.0
- inert@3.2.0

When using MySQL server the following module is also required:

- mysql@2.11.1

When using MSSQL server the following modules are also required:

- tedious@1.12.3
- tedious-connection-pool@0.3.9

When using the Google reCaptcha the following module is also required:

- request@2.69.0

**Upload folder**
When using blob fields, a specific folder is required where files can be uploaded before they are inserted in a blob field of the database. Note that uploaded files are deleted automatically after they have been inserted in the database.

**MySQL**

- When using Linux:

  Create a subfolder called “uploads” in the following folder: “/var/lib/mysql/”

  Make sure the MySQL server has full ownership and permissions for the files in the “uploads” folder by executing the following commands:

  - “chown mysql:mysql /var/lib/mysql/uploads/”
  - “chmod go+rw /var/lib/mysql/uploads/”

- When using Windows:

Make sure a folder called “uploads” exists in the following folder:
“/ProgramData/MySQL/MySQL Server 5.7/”

**MSSQL**

Create a subfolder called “uploads” in the myCloudData SDK install folder.

**The API Configuration settings**

The API settings can be found in the file called “cfg.js”.

Configuration file structure:

- **masterPassword (string):**
  This is the master password which is used to encrypt the user passwords in the database. It is highly recommended to change the default value before adding users to the system (including admin account initialization described below).
  Note: the value of the master password cannot be changed after one or more users have been added to the system otherwise the user passwords will become invalid and it will no longer be possible to login or authenticate.

- **isLocal (boolean):**
  Indicates if the server will run local or remote. If remote, the server will only allow strictly control panel related API calls from the domain value as indicated in the “domain” property.

- **domain (string):**
  Defines the domain host name for remote servers. Only required if isLocal is set to false.

- **apivhost (string array):**
  Defines the host names for which the API must be accessible.

- **sitevhost (string array):**
  Defines the host names for which the control panel website must be accessible.

- **database (string):**
  Defines which type of database connection will be used. Accepted values are “MSSQL” or “MYSQL”.

- **uploadPath (string):**
  Defines the path where files can be uploaded before they are inserted in a blob field of the database.
  Note: uploaded files are deleted automatically after they have been inserted in the database.

- **server:**

  - **enableHttp (boolean):**
    Indicates if the server will be accessible via HTTP.

  - **enableHttps (boolean):**
    Indicates if the server will be accessible via HTTPS.

  - **portHttp (integer):**
    Defines the port number to use for HTTP access. Default value is 80.

  - **portHttps (integer):**
    Defines the port number to use for HTTPS access. Default value is 443.

  - **hostHttps (string):**
    The IP address where the server is running on. Only required if enableHttps is true.

- **tls:**
  The required files to enable a valid and secure HTTPS connection. Only required if enableHttps is true.

  - **key (string):**
    Path and filename where the “privatekey.key” file is located.

  - **cert (string):**
    Path and filename where the domain certificate file is located.

  - **ca (string array):**
    Path and filenames where the certificate authority files are located.

- **mssql:**
  The MS SQL database connection settings. Only required if a MS SQL database server is used.

  - **userName (string):**
    Defines the username for the database connection.

  - **password (string):**
    Defines the password for the database connection.

  - **server (string):**
    Defines the server for the database connection. Default value is “localhost”.

- **mysql:**
  The MySQL database connection settings. Only required if a MySQL database server is used.

  - **userName (string):**
    Defines the username for the database connection.

  - **password (string):**
    Defines the password for the database connection.

  - **server (string):**
    Defines the server for the database connection. Default value is “localhost”.

  - **database (string):**
    Defines the database name for the database connection.

- **captcha:**
  The captcha settings. Only required if the control panel is configured to display a captcha on the form of the “Account Details” page.

  - **enable (boolean):**
    Indicates if the serverside captcha check should be enabled.

  - **secret (string):**
    Defines the “secret” value associated with the captcha. This value must be obtained from the Google reCaptcha control panel.

**The Control Panel configuration settings**

The API settings can be found in the file called “site/assets/scripts/config.js”.

Configuration file structure:

- **APIBase (string):**
  The base URL value (including port number) where the API can be accessed. The default value is “http://localhost/”. Note that if the portHttp or portHttps values are different from their default value, the port number should also be included in this value.

  Example:
  “http://localhost:8888/”

- **captcha:**
  The captcha settings. Note that in the API configuration settings the captcha should be enabled and a valid secret value should be provided if the captcha is enabled here.

  - **enable (boolean):**
    Indicates if the captcha is displayed on the form of the “Account Details” page.

  - **key (string):**
    Defines the “key” value associated with the captcha. This value must be obtained from the Google reCaptcha control panel.

## Starting the server application

Navigate to the myCloudData SDK folder and execute the following command:

"node server.js"

If the server start was successful a message will appear that the server is running.

## Initialize the default admin account

The default admin account allows to manage (add, update, delete) the users who have access to the myCloudData control panel and API.

To automatically generate the default admin account navigate to the following endpoint in your browser:

“mydomain:portnumber/v2/init”

If the initialization was successfull you are redirected to the control panel page, if not, an error message is displayed.

The default admin account credentials are:

- Login/email: mycloudddata@mycloudddata.net
- Password: password

It is highly recommended to change the admin password after logging in to the control panel.
To change the admin password:

- Navigate to the control panel in your browser
- Log in with the default admin user credentials
- Click the “Manage your tables” button
- Click the “ACCOUNT DETAILS” menu item
- The admin user account details are displayed

- Enter the default password in the “Old Password” field
- Enter the new password in the “New Password” and “Confirm Password” fields
- Update the admin account details

Now you are ready to start using the myCloudData SDK.

- To open the control panel in your browser navigate to:

  “mydomain:portnumber”
- To test if the API is running in your browser navigate to:

  “mydomain:portnumber/version”

  If the API version number is displayed, the API server is running as expected.