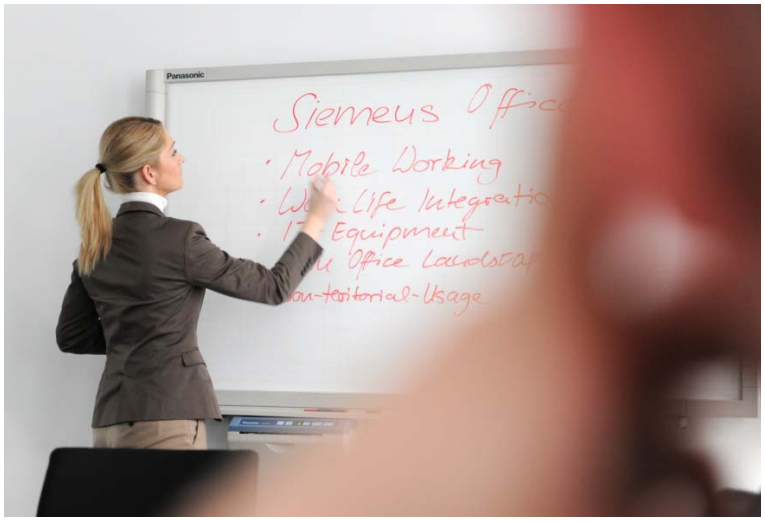
SIEMENS



MM8000 V4.70

REST Web API for Event Handling

Programming Guide

Building Technologies

Fire Safety & Security Products

1 Introduction

This document describes the MM8000 Web Services interface (REST Web API) for handling MM8000 events from any client supporting this Representational State Transfer (REST) Application Program Interface (API).

The lists of REST API methods exposed by the Web Services are grouped by controller name.
The supported controllers are the following:

- Tables
- Diagnostics
- Images
- Login
- Languages
- Events

For each group, the following sections present the details of the methods.

2 REST API: Tables

Request URL:http:/.../api	Short description	Request Method	Parameter (optional / mandatory)	Body Response
Tables				

/Tables/Version
 	
this method is also implemented with url: **http:/.../api/Diagnostics**
	Get the server version and the interval of compatible client versions	GET		`{`
				` Server_Major_Version: <y>,`
				` Server_Minor_Version: <x>,`
				` Min_ClientApp_Major_Version: <Y>,`
				` Min_ClientApp_Minor_Version: <y>,`
				` Authentication: <string(basic|form|openid|oaut)>,`
				` ServerId": xyz`
				`}`

/Tables/Categories	Get list of the categories	GET		Dictionary of categories Dictionary<int, string>
/Tables/Categories/Subtables/Colors	Get list of the lcolor/list of each category	GET		Dictionary with color-list per category Dictionary<int, Dictionary<int,List<int>>>
/Tables/Disciplines	Get list of the disciplines	GET		Dictionary of disciplines Dictionary<int, string>
/Tables/Disciplines/Subtables/Icons	Get list of icons of each discipline	GET		Dictionary with icon per discipline Dictionary<int, string>

/Tables/Getcurrentuser	Get current user LCID, Login code and ServerId	GET		`{`
				` success: <boolean>,`
				` data: {`
				` Login: <string>,`
				` LCID: <integer>,`
				` ServerId: <string>`
				` },`
				` extraInfo: <string>,`
				` code: <integer>`
				`}`

2.1 Version

This method is the first called and it checks if the server is available through Anonymous connection.

2.2 Categories

This method gets the *Categories* of the events. The response is a list of *Category Id* and *Category description* in the current language of the user:

Response example:

```
"1": "Advisory",
"2": "Anomaly",
"3": "non Default Mode",
"4": "Exclusion",...
```

2.3 Categories/Subtables/Colors

This method gets the *Colors* of the *Categories* of the events. The response is a list of a 13 colors of each category. Each color is defined by the *RgbaCode* in a comma-separated string. The 13 values are different for the different cases in which the color has to be displayed (button blinking, button pressed, and so on).

Response example:

```
"1": {
"1": "32,140,188",
"2": "32,140,188",
"3": "32,140,188",
"4": "32,140,188",
"5": "32,140,188",
"6": "32,140,188",
"7": "32,140,188",
"8": "32,140,188",
"9": "32,140,188",
"10": "32,140,188",
"11": "32,140,188",
"12": "32,140,188",
"13": "32,140,188"
},
...
```

2.4 Disciplines

This method gets the *Disciplines* of the events. The response is a list of *Discipline Id* and *Discipline description* in the current language of the user:

Response example:

```
"1": "Fire",
"2": "Intrusion",
"3": "Hardware",
"4": "Building Services",
"5": "Access Control",
"6": "System Services",
"7": "Management",
"8": "Input Output",
"9": "Gas",
"10": "CCTV"
```

2.5 Disciplines/Subtables/Icons

This method gets the name of the icons of the each discipline preceded by the path in which the icons are located.

Response example:

```
"0": "",
"1": "Resources\\images\\Disciplines\\dis-fire",
"2": "Resources\\images\\Disciplines\\dis-intrusion",
"3": "Resources\\images\\Disciplines\\dis-hardware",
"4": "Resources\\images\\Disciplines\\dis-building",
"5": "Resources\\images\\Disciplines\\dis-access-control",
"6": "Resources\\images\\Disciplines\\dis-system-service",
```

"7": "Resources\\images\\Disciplines\\dis-energy-management",
"8": "Resources\\images\\Disciplines\\dis-input-output",
"9": "Resources\\images\\Disciplines\\dis-gas",
"10": "Resources\\images\\Disciplines\\dis-cctv"

2.6 Getcurrentuser

This method gets current user *LCID*, *Login code* and *Server Id*.

- Login code: Login of the user
- LCID: Language code ID of the user.
- ServerId: is a unique identifier of the server.
  This value is used to filter the android push notification correctly.

Response example:

```
success: true,
 "data": {
 "Login": <login code>,
 "LCID": 1033,
 "ServerId" : <ServerId>
 }
```

3 REST API: Diagnostics

Request URL:http:/.../api	Short description	Request Method	Parameter (optional / mandatory)	Body Response
Diagnostics				
/Diagnostics	Get server version, server ID and the interval of compatible client versions	GET		{ Server_Major_Version: <y>, Server_Minor_Version: <x>, Min_ClientApp_Major_Version: <Y>, Min_ClientApp_Minor_Version: <y>, Authentication: <string(basic|form|openid|oaut)>, ServerId": <xyz> }

4 REST API: Images

Request URL:http:/.../api	Short description	Request Method	Parameter (optional / mandatory)	Body Response
Images				
/Images/{imagename}	Get 64 base code of an image	GET	**m** string imagename **o** format=<image format> **o** width=<image width> **o** height=<image height> **o** encodedAs-Base64=True **o** path=<image server path>	String that represents 64 base code of the image

4.1 Images

This method gets the 64 base code of an image.

This method is called for all the discipline icons and it is called 4 times for the 4 fixed images that represent the command icons:

1. ack-black
2. ack-white
3. reset-black
4. reset-white

5 REST API: Login

Request URL:http:/.../api	Request URL:http:/.../api	Short description	Body Response
Login			
/Login/Login	Login the user	POST	int connection_state (see enum below)
/Login/RegisterDevice	Subscribe for push notification	POST	Token: <string> Name: <string> UUID: <string> Platform: <string>
/Login/Logout	Logout the user	POST	

5.1 Login

This method uses the *WebSecurity* class that provides security and authentication features for *ASP.NET Web Pages* applications, including the ability to create user accounts, log users in and out, reset or change passwords, and perform related tasks.

When a user is logged in, ASP.NET sets an authentication token in a cookie that lets ASP.NET know on subsequent requests that the user has been logged in.

The method checks if the user credentials are correct, and in this case it opens a WebSecurity session for the user.

If this method succeeds, it returns 0 (*Connected*); other possible values are defined below:

connection_state enumeration

```
Connecting = 1,
Connected = 0,
NotLogged = -1,
ServerNotConnected = -2,
InvalidCredential = -3,
ServicesNotAvaible = -4,
AccountLocked = -5,
InvalidLicense = -6,
ProjectDownloading = -7,
Unknown = -100
```

5.2 RegisterDevice

This method is used for enable the push notification. If this method succeeds, it returns a JSON object as shown here below.

Response example:

Token: aaa
Name: bbb
UUID: ccc
Platform: ddd

5.3 Logout

This method calls the *WebSecurity* class to close the user session trough the *WebSecurity.Logout()* that removes the authentication token, with the effect of logging the user out.

6 REST API: Languages

Request URL:http:/.../api	Short description	Request Method	Parameter (optional / mandatory)	Body Response
Languages				
/Languages/GetLanguages	Get all the languages supported by the application	GET	**o** int page **o** int size **o** int start **o** int limit	language_object as JSON (see below)
/Languages/GetLanguage	Get all the texts in a specified language	GET	**m**: string language_object (see below)	JSON language_texts
/Languages/SetLanguage	Set the language to the current user	POST	**m**:language_object as JSON (see below)	

All these methods use the language object as JSON defined here below:

language_object

```
int LCID
String Description
String Code
String Texts in JSON format
```

A language_object example is:

```
"LCID":1033,
"Description":"English",
"Code":"en",
"Texts":""event": {"acknowledge": "Acknowledge",...},"filters": {"yesterday":
"Yesterday","commands\": \"Commands",..."
```

6.1 GetLanguages

This method gets all the languages supported by the application. It returns them as the list of language objects with null Texts data for performance reason.

Response example:

{"LCID":1033,"Description":"English","Code":"en","Texts":null},
{"LCID":1040,"Description":"Italiano","Code":"it","Texts":null}]

6.2 GetLanguage

This method gets all the texts for the application of a language, which has to be specified as LCID value in the parameter string.

The response is a JSON object with all the texts needed by the application to be localized.

Parameter example for setting English to current user :

"Description":"","Code":"","LCID":1033,"Texts":""

Response example:

"event": {
 "acknowledge": "Acknowledge",
 ...
 },
 "filters": {
 "today": "Today",
 "commands\": \"Commands",
 ...

6.3 SetLanguage

This method sets in DB the language to the current user. All devices of the user has the same language.

7 REST API: Events

Request URL:http:/.../api	Short description	Request Method	Parameter (optional / mandatory)	Body	Response
Events					
/Events	Get the event list	POST	**o** int Total **o** int Page **o** int Start **o** int Size	**o** Filters	See details below
/Events/{eventid}/Commands/{commandId}	Execute a command of an event	POST	**m** string eventId **m** int commandId		

7.1 Events

This method gets all the events present in the system according to the specified filters.

Response example:

```
"Events": [... ],
 "info": {
 "CategoriesCount": [
 {
 "CategoryId": 1,
 "Count": 0,
 "AcknowledgeCount": 0,
 "ResetCount": 0
 },
 {
 "CategoryId": 2,
 "Count": 0,
 "AcknowledgeCount": 0,
 "ResetCount": 0
 },
...
```

The Events list is defined as follows:

Name	Type	Description	Mand./ Optional
Id	string	MM8000 alarm identifier	M
Deleted	bool	Flag whether an event has been deleted. This is for push notifications only.	O
EventId	uint	Event counter (non-unique)	M
CategoryId	int	Alarm category	M
CategoryName	string	Localized textual representation of CategoryId	O
StateId	Int	Alarm state (consider to redefine to encapsulate the active/inactive source state)	M
StateName	String	Localized textual representation of StateId	O

Commands	List of available commands	List to available commands (id, link, name) Reset, Ack	M
Cause	string	Cause came with alarm (localized)	M
SrcSystemId	int	Unique system identifier	M
SrcViewName	string	View name	M
SrcViewDescription	string	View description	M
SrcDesignation	string	Concatenated tree-node names	M
SrcLocation	string	Concatenated tree-node descriptions	M
SrcName	string	Tree-node name	
SrcDescription	string	Tree-node description	
SrcDisciplineId	int	Alarm discipline	M
SrcDisciplineName	string	Localized textual representation of DisciplineId	O
CreationTime	DateTime	Time when alarm has been created	M
DirectionId	int	Indicates IN or OUT alarm	M
DirectionName	string	Localized textual representation of DirectionId	O
_links	List of Link	List of available links such links are used for example to refer to the related Resources such a Command Table Resource	O

Command is defined as below:

Name	**Type**	**Description**	**Mand./ Optional**
Id	Int	Command Id (unique within event)	M
EventId	string	Foreign key to event	M
Name	string	Localized command name according to session	M
_links	List of Link	List of available links (e.g. to itself for executing command)	O

Link is defined as below:

Name	**Type**	**Description**	**Mand./ Optional**
Href	string	Hyperlink to a resource	M
Rel	string	Link relation types	M

Href is like this:

```
api/events/" + {EventId} + "/commands/" + {command_state}
```

where *command_state* can be *unreset state* or *unacknowledged state*.

Filters

The events can be filtered by:

1. CreationTime
2. States
3. Categories
4. Disciplines

CreationTime possible values are:

1 Events generated in last 15 minutes
2 Events generated in last 30 minutes
3 Events generated in last hour
4 Events generated last night
5 Events generated yesterday
6 Events generated today

CreationTime parameter is a single value of the required time value.

State values:

1 Events in unreset state
2 Events in unacknowledged state

States parameter is a list of State values to be filtered.

Disciplines parameter is a list of DisciplineId to be filtered.

Categories parameter is a list of CategoryId to be filtered.

7.2 Events/{eventid}/Commands/{commandId}

This method executes a command on events. Clients can use the first link of the available list of links in the command, and execute its Href.

For example, if an event has the following command list:

"Commands": [

```
{
"_links": [
{
"Rel": "self",
"Href": "api/events/2/commands/xxxx",
},
{
"Rel": "self",
"Href": "api/events/2/commands/yyyy",
}
]...
```

The "api/events/2/commands/xxxx" "Href":will be used to execute the command.

8 Appendix

8.1 UI Labels


```
{
    "server": {
        "pushNotification": {
            "deviceRemovedForInactivity": "This device has been removed for inactivity."
        }
    },
    "event": {
        "acknowledge": "Acknowledge",
        "sendingCommandError": "Error in sending the command.",
        "discipline": "Discipline",
        "location": "Location",
        "cause": "Cause",
        "error": "Error:",
        "reset": "Reset",
        "date": "Date",
        "time": "Time",
        "additionalInformation": "Additional information",
        "eventClosed": "Event closed",
        "noCommands": "No commands",
        "eventView": "Event view",
        "id": "ID"
    },
    "filters": {
        "yesterday": "Yesterday",
        "commands": "Commands",
        "last30minutes": "Last 30 minutes",
        "last15minutes": "Last 15 minutes",
        "reset": "Reset",
        "today": "Today",
        "discipline": "Discipline",
        "clear": "Clear",
        "lastHour": "Last hour",
        "filters": "Filters",
        "acknowledge": "Acknowledge",
        "eventDate": "Event date",
        "lastNight": "Last night"
    },
    "signIn": {
        "domain": "Domain",
        "registerDeviceError": "The push notification server is temporarily not available. Please, try to enable the
notification later through Settings menu.",
        "invalidCredentials": "Invalid credentials",
        "password": "Password",
        "serverUrlExample": "Server url with http://",
        "urlInvalid": "Invalid url",
        "signIn": "Sign In",
        "fieldsRequired": "Fields are required"
    },
    "errorCode": {
        "serverNotAvailable": "Server not available",
        "projectDownloading": "Downloading project",
        "unknowError": "Unknown error",
        "invalidCredential": "Invalid credentials",
        "accountLocked": "Your user is locked, logout and try to reconnect to the app",
        "connecting": "The server is connecting",
        "invalidLicence": "Invalid licence",
        "notLogged": "The user is not logged"
    },
    "misc": {
        "btnGotoLogin": "Go to login",
        "userNotFound": "User not found",
        "confirmPin": "Confirm Pin",
        "askClose": "Do you really want to close the application?",
        "deviceRegistration": "Device registration",
        "genericConfigurationError": "Error during loading the configuration.",
        "waitLoading": "Loading data...",
        "loginErrorText": "Impossible to establish the connection with the server.",
        "loadLocaleMsg": "Loading...",
        "btnAccept": "Accept",
        "logout": "Logout",
        "error": "Error",
        "close": "Close",
        "noEvents": "No events",
        "serverDisconnected": "Server is disconnected, try to reconnect or contact your admin.",
        "categories": "Categories",
        "pin": "Pin",
        "all": "All",
        "cancel": "Cancel",
        "btnChangeLoginAccount": "Change account",
        "deviceRooted": "The device is rooted, the application can not start.",
        "disciplines": "Disciplines",
        "seconds": "seconds",
        "noInternetConnection": "No Internet connection",
```

```
        "internetConnectionRequired": "Please check your connection and try again.",
        "settings": "Settings",
        "unknown": "Unknown",
        "waitLogin": "Login...",
        "tapToRefresh": "Tap to update",
        "btnRestartConnection": "Restart connection",
        "waitConnecting": "Connecting to server...",
        "noLongerUpdated": "Updates available",
        "user": "User",
        "wait": "Wait...",
        "none": "None"
    },
    "header": {
        "description": "English",
        "code": "en",
        "LCID": "1033"
    },
    "button": {
        "apply": "Apply",
        "ok": "Ok",
        "ignore": "Ignore",
        "cancel": "Cancel"
    },
    "options": {
        "serverVersion": "Server version",
        "registerDeviceError": "The push notification server is temporarily not available. Please, try to enable the
notification later.",
        "buttons": {
            "server": "Server"
        },
        "titles": {
            "settings": "Other Options",
            "pin": "Security",
            "privacyPolicy": "Privacy Policy",
            "termsOfUse": "Terms of Use",
            "languages": "Application languages",
            "corporateInformation": "Corporate Information",
            "notificationsFilter": "Notifications"
        },
        "appVersion": "App version"
    },
    "mail": {
        "source": "Source",
        "category": "Category",
        "date": "Date",
        "time": "Time"
    },
    "pin": {
        "pin": "Pin",
        "confirmNewPin": "Confirm new Pin",
        "newPin": "New Pin",
        "enablePin": "Enable Pin",
        "invalidCurrentPin": "Current Pin is invalid",
        "invalidPin": "Invalid Pin",
        "configurePin": "Pin",
        "insertNewPinandConfirmPin": "New and Confirm Pin have to follows the policies and match",
        "askconfigurePin": "A pin code can be configured from the settings menu, do you want to set it now?",
        "rulesInfo": "Insert only numbers; min 6 digits and max 20 digits",
        "currentPin": "Current Pin",
        "insertPin": "Insert Pin",
        "differentNewPinandConfirmPin": "New Pin is different from Confirm new Pin",
        "changeAccount": "Logout"
    },
    "connSettings": {
        "reconnectIntervall": "Reconnect interval",
        "connectionTimeout": "Connection timeout",
        "reconnectAutomatically": "Reconnect automatically"
    },
    "eventlist": {
        "list": {
            "pullrefreshplugin": {
                "releaseText": "Release to refresh...",
                "loadedText": "Loaded",
                "lastUpdatedText": "Last Updated:",
                "loadingText": "Loading...",
                "pullText": "Pull down to refresh..."
            },
            "pagingplugin": {
                "nomorerecordstext": "No more events",
                "loadmoretext": "Load more..."
            }
        },
        "eventList": "Event list"
    },
    "share": {
        "shareVia": "Share Via",
        "email": "Email",
        "sms": "Sms"
    },
    "buttons": {
        "done": "Done",
        "cancel": "Cancel"
```

```
    },
    "months": {
        "1": "January",
        "2": "February",
        "3": "March",
        "4": "April",
        "5": "May",
        "6": "June",
        "7": "July",
        "8": "August",
        "9": "September",
        "10": "October",
        "11": "November",
        "12": "December"
    }
}
```

Issued by
Siemens Switzerland Ltd
Infrastructure & Cities Sector
Building Technologies Division
International Headquarters
Gubelstrasse 22
CH-6301 Zug
Tel. +41 41-724 24 24
www.siemens.com/buildingtechnologies