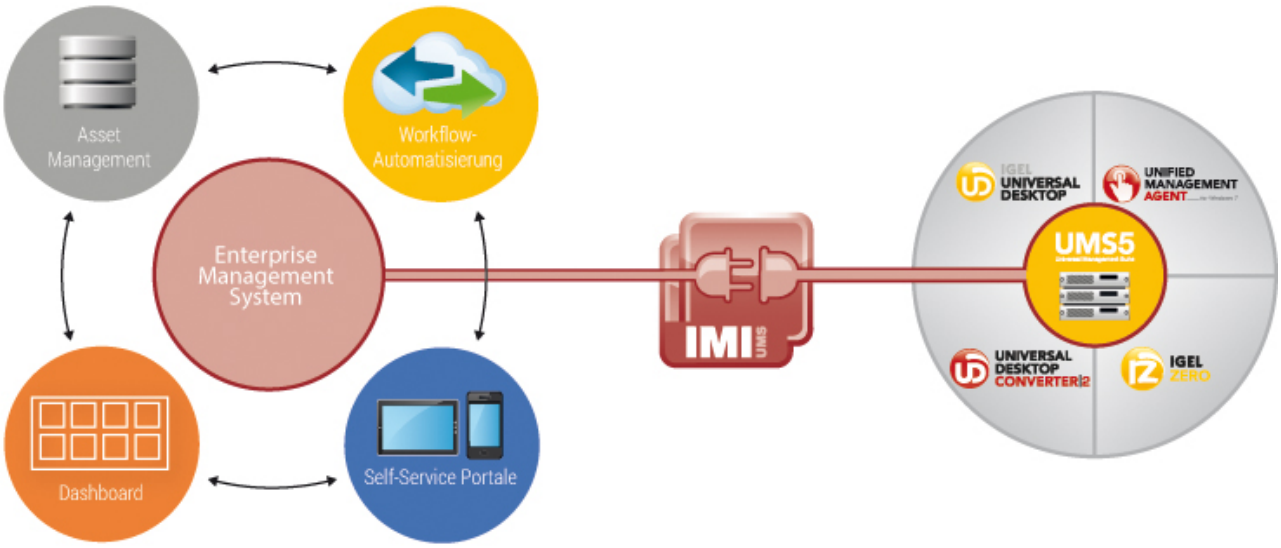
IGEL Management Interface (IMI)

- IMI Manual (see page 3)
- Powershell (see page 38)
- IMI API V1 Reference (see page 90)

IMI Manual

IGEL Management Interface (IMI) enables you to connect UMS to systems management tools. It is a programming interface that can create and delete thin clients, move them between directories, reboot them and much more. Its implementation as a REST API makes IMI agnostic of hardware platforms, operating systems and programming languages, thus ensuring maximum interoperability.



This document serves as an introduction to using IGEL Management Interface (IMI).

Detailed information about the requests can be found in the IMI API V1 Reference (see page 90).

- Licensing (see page 4)
- REST Basics (see page 5)
- Prerequisites (see page 8)
- First Steps (see page 9)
- Creating, Updating and Deleting Resources (see page 25)
- Further Operations (see page 33)

Licensing

The licensing for IGEL Management Interface (IMI) depends on the UMS version you are using.

When Asset Inventory Tracker (AIT) has been licensed, you can use the resources `assetinfo` and `assethistory`.

UMS 5.09.100 or Older

For basic functionality, see Licensing IMI.

For Asset Inventory Tracker (AIT), see Licensing AIT.

UMS 6.01.100 or Newer

IMI is part of the IGEL Workspace Edition (WE); no additional license is necessary. For further information, see Workspace Edition.

For Asset Inventory Tracker (AIT), licenses from the Enterprise Management Pack (EMP) are required. For further information, see Enterprise Management Pack.

REST Basics

IGEL Management Interface uses *REST,* which stands for Representational State Transfer.

REST is an architectural style for client-server applications, mainly implemented in the HTTP(S) protocol. Therefore it can be used with all technologies that can send and receive HTTP requests.

REST establishes a typical pattern that helps programmers to understand the structure of individual APIs. Its most important concepts are Ressources (see page 6) and HTTP Methods (see page 7).

Ressources

URLs Represent Resources

A *REST* API makes resources available at specific URLs. You can find a list of all thin clients in the UMS REST API at:

- `https://[server]:8443/umsapi/v3/thinclients`

In a shorter notation, which assumes the base URL to be known:

- `/v3/thinclients`

In order to address an individual instance of a resource - a thin client,for example - you specify its ID in the URL:

- `/v3/thinclients/8`

Further Examples of Resources

URL	Resource
`/v3/directories/tcdirectories`	A list of all thin client directories
`/v3/directories/tcdirectories/123`	The thin client directory with the ID 123
`/v3/firmwares`	A list of all firmwares in the UMS instance
`/v3/firmwares/7`	The firmware with the ID 7

Find a list of all available resources in the IMI API Reference (see page 90).

HTTP Methods

You call HTTP methods for resources in order to manipulate them. The REST architectural style has a conventional meaning for each of the HTTP methods, which are also called verbs. The methods are listed below:

HTTP Method	
`GET`	Read information from a resource.
`PUT`	Create a new resource or update an existing resource.
`POST`	(Create a new resource *), send a command.
`DELETE`	Delete a resource.

* There is a subtle semantic difference between PUT and POST, which is sometimes a matter of dispute. *IMI* API favors

- PUT for create and update actions and uses
- POST for logins and for sending commands to resources.

Prerequisites

IGEL Universal Management Suite (UMS)

UMS 5.01.100

Networking

In order to use *IGEL Management Interface* you need to be able to reach the API host via the network and connect to its API port, TCP 8443 by default.

The base URL is

- `https://[server]:8443/umsapi/`

The resources for IMI version 1 are available at:

- `https://[server]:8443/umsapi/v1/`

> *IMI* uses HTTPS to ensure the integrity and confidentiality of the network traffic. It is good practice to use a valid server certificate with a verifiable signature. Most clients and libraries can be configured to work with self-signed or invalid certificates, but that should not be done in a production environment.

First Steps

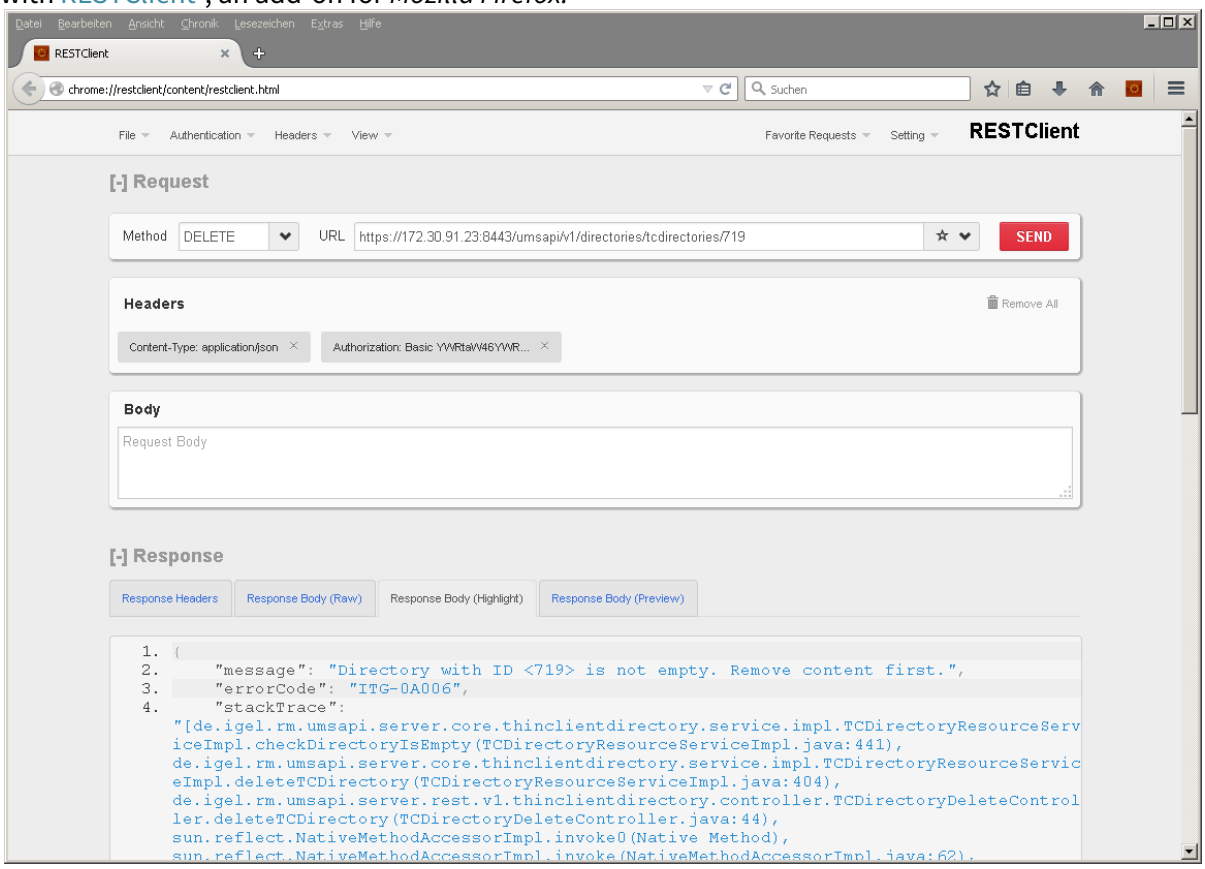
- Client Applications and Libraries (see page 10)
- Getting the Server Status (see page 12)
- Authentication (see page 14)
- Listing all Thin Clients (see page 15)
- Getting Information on a Thin Client (see page 18)
- Getting All Details about a Thin Client (see page 20)
- Getting Thin Client Status (see page 23)

Client Applications and Libraries

Clients

The easiest way to try out the *IGEL Management Interface* is either

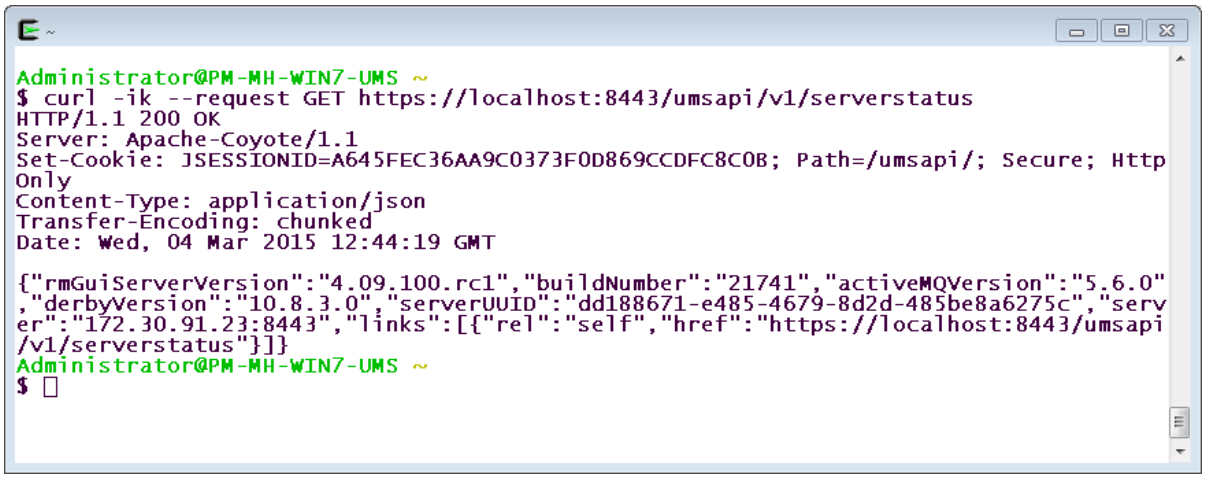
- with RESTClient[1], an add-on for *Mozilla Firefox:*



or

- with cURL[2], a commandline network client:

1 http://restclient.net/
2 http://curl.haxx.se/



This guide uses cURL for examples, as the commandline makes all parameters visible in plain text. This need not stop you from using RESTClient if you are more comfortable with it. You can easily translate the commandline parameters into the fields of the RESTClient *RESTClient* GUI.

Programming libraries

Most programming languages provide an HTTP and an SSL module, either in their standard library or as an extension, and a JSON library (for the API data format) as well.

Getting the Server Status

▶ Send the endpoint `/v3/serverstatus` an HTTP GET request to retrieve some information about the server.

With *cURL* the commandline looks like this:

```
curl \
--request GET \
https://[server]:8443/umsapi/v3/serverstatus
```

You will receive a response like the following:

```
200 OK
---
{
 "rmGuiServerVersion": "5.07.100.rc15",
 "buildNumber": "32287",
 "activeMQVersion": "5.6.0",
 "derbyVersion": "10.12.1.1",
 "serverUUID": "9fe719f1-c16e-4744-9ff1-b9c314ae151c",
 "server": "mhuber:8443",
 "links": []
}
```

Response Headers

The server replied with the status code 200 in the HTTP header, which means that your request was successful.

You can find a list of status codes in the IMI API Reference (see page 90).

JSON Response Body

Information about the server is given in the response body in the *JavaScript Object Notation (JSON)*format. It is enclosed in curly braces and contains key-value-pairs separated by a colon. The line separator is a comma. Your

programming language probably offers a `JSON` module that will help you parse and process information in this format.

Authentication

You must be authenticated in order to use *IGEL Management Interface,* otherwise the server will return the HTTP status 401 "Unauthorized". Only querying the server status is allowed without authentication.

IMI uses *Basic Authentication* (RFC 2617 (see page 14)).

Logging In

▶ Send an HTTP POST request to `/v2/login` to log in. Your HTTP client of choice will offer a way of sending a *Basic Authentication-* header, which is produced from the username/password combination.

This is the commandline for *cURL:*

```
curl \
--request POST \
--user '[Username]:[Passwort]' \
https://[server]:8443/umsapi/v3/login
```

The server replies, sending back a session ID:

```
200 OK
Set-Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC; Path=/umsapi/; Secure; HttpOnly
---
{"message":"JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC"}
```

Staying Logged in

▶ You can maintain the session by sending the JSESSIONID in the Cookie header with every subsequent request:

```
Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC
```

Some clients will do this automatically for you, e.g. the RESTClient (see page 10)Firefox add-on (see page 10).

Listing all Thin Clients

- Send a GET request to `/v3/thinclients` to list all thin clients.
- Don't forget to tell the server your JSESSIONID:

```
curl \
--request GET\
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/thinclients
```

Response

The response contains a large *JSON* document, listing all thin clients and their most important properties. The links make further parts of the API discoverable for a client that spiders the interface.

Look up the format for a thin client resource in the IMI API Reference (see page 90).

The last thin client entry contains `"movedToBin": true`, meaning that the thin client has been deleted and moved to the **Recycling Bin**. Objects in the **Recycling Bin** are listed, but you cannot update them or send them commands.

```
[
 {
 "unitID": "00E0C54DCB8E",
 "mac": "00E0C54DCB8E",
 "firmwareID": "21",
 "lastIP": "172.30.91.43",
 "id": "27",
 "name": "Front Desk",
 "parentID": "-1",
 "movedToBin": false,
 "objectType": "tc",
 "links": [
```

```
{
"rel": "self",
"href": "https://172.30.91.227:8443/umsapi/v3/thinclients/27"
},
{
"rel": "Parent",
"href": "root"
},
{
"rel": "Firmware",
"href": "https://172.30.91.227:8443/umsapi/v3/firmwares/21"
}
]
},
{
"unitID": "DC9C5207694E",
"mac": "DC9C5207694E",
"firmwareID": "13",
"lastIP": "172.30.91.24",
"id": "6888",
"name": "UD3 M340C_Board",
"parentID": "15592",
"movedToBin": false,
"objectType": "tc",
"links": [...]
},
```

```
 {
 "unitID": "00E0C5080834",
 "mac": "00E0C5080834",
 "firmwareID": "2",
 "lastIP": "172.30.91.132",
 "id": "6899",
 "name": "UD10",
 "parentID": "15592",
 "movedToBin": true,
 "objectType": "tc",
 "links": [...]
 }
]
```

Getting Information on a Thin Client

▶ Send a GET request to `/v3/thinclients/[id]` to get information about a thin client:

```
curl \
--request GET\
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/thinclients/27
```

The thin client ID is passed to the API as a *Path Variable* in the URL.

The response contains the most important properties of the thin client with the ID 27:

Response


```
{
 "unitID": "00E0C54DCB8E",
 "mac": "00E0C54DCB8E",
 "firmwareID": "21",
 "lastIP": "172.30.91.43",
 "id": "27",
 "name": "Front Desk",
 "parentID": "-1",
 "movedToBin": false,
 "objectType": "tc",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.227:8443/umsapi/v3/thinclients/27"
 },
 {
```

```
 "rel": "Parent",
 "href": "root"
 },
 {
 "rel": "Firmware",
 "href": "https://172.30.91.227:8443/umsapi/v3/firmwares/21"
 }
 ]
}
```

Getting All Details about a Thin Client

▶ Use the `details` facet in order to get all details about a thin client:

```
curl \
--request GET\
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/thinclients/27?facets=details
```

The response contains alle the properties of the thin client with the ID 27:

Response

```
{
 "unitID": "00E0C54DCB8E",
 "mac": "00E0C54DCB8E",
 "firmwareID": "21",
 "networkName": "D.Weinert",
 "site": "1. Stock",
 "department": "Product Management",
 "lastIP": "172.30.91.43",
 "costCenter": "",
 "comment": "",
 "assetID": "",
 "inserviceDate": "01.06.215",
 "serialNumber": "",
 "productId": "UD6-LX 51cps",
 "umsStructuralTag": "",
 "cpuSpeed": 2416,
 "cpuType": "Intel(R) Celeron(R) CPU J1900 @ 1.99GHz",
```

```
"deviceType": "IGEL H830C",
"deviceSerialNumber": "14D3D3C03B14470B9EM",
"osType": "IGEL Linux V5 (Kernel Version 3.13.11-ckt20)",
"flashSize": 1883,
"memorySize": 1853,
"networkSpeed": 1000,
"graphicsChipset0": "INTEL HD Graphics (Baytrail)",
"graphicsChipset1": "",
"monitorVendor1": "Samsung Electric Company",
"monitorModel1": "S24C650",
"monitorSerialnumber1": "H4MG404381",
"monitorSize1": 24,
"monitorNativeResolution1": "1920 x 1200",
"monitor1YearOfManufacture": "2015",
"monitor1WeekOfManufacture": "17",
"monitorVendor2": "Samsung Electric Company",
"monitorModel2": "S24C650",
"monitorSerialnumber2": "H4MG404389",
"monitorSize2": 24,
"monitorNativeResolution2": "1920 x 1200",
"monitor2YearOfManufacture": "2015",
"monitor2WeekOfManufacture": "17",
"biosVendor": "INSYDE Corp.",
"biosVersion": "H830C V:3.5.13-11282014",
"biosDate": "11/28/2014",
"totalUsagetime": "5798368000",
```

```
 "totalUptime": "18513000",
 "lastBoottime": "2015-09-29 08:30",
 "id": "27",
 "name": "Front Desk",
 "parentID": "-1",
 "movedToBin": false,
 "objectType": "tc",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.227:8443/umsapi/v3/thinclients/27"
 },
 {
 "rel": "Parent",
 "href": "root"
 },
 {
 "rel": "Firmware",
 "href": "https://172.30.91.227:8443/umsapi/v3/firmwares/21"
 }
 ]
}
```

Getting Thin Client Status

▶ Use the `online` facet in order to get the online status of a thin client with a specific ID:

```
curl \
--request GET\
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/thinclients/27?facets=online
```

The response contains, among others, the `online` property, which can have the values `true` and `false`:

Response

```
{
 "unitID": "00E0C54DCB8E",
 "mac": "00E0C54DCB8E",
 "firmwareID": "21",
 "lastIP": "172.30.91.43",
 "online": false,
 "id": "27",
 "name": "Front Desk",
 "parentID": "-1",
 "movedToBin": false,
 "objectType": "tc",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.227:8443/umsapi/v3/thinclients/27"
 },
 {
```

```
 "rel": "Parent",
 "href": "root"
 },
 {
 "rel": "Firmware",
 "href": "https://172.30.91.227:8443/umsapi/v3/firmwares/21"
 }
 ]
}
```

Creating, Updating and Deleting Resources

The thin client directories in UMS are a good resource for trying out the creation, update and deletion API calls.

Listing all Thin Client Directories

▶ List all thin client directories:

```
curl \
--request GET\
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/directories/tcdirectories/
```

Response

The response contains a list of thin client directories in *JSON* format. The ID of the root thin client directory is always -1. Find a detailed description of the **Thin Client Directory** resource in the IMI API Reference (see page 90).

▶ Compare the *JSON* response to the thin client directory tree you see in your *UMS console*.

IMI provides a flat view of these directories, subdirectories can only be recognized via their `parentID`. In order to better see the nesting, use the `children` facet.

```
[
 {
 "id": "15592",
 "name": "Pool",
 "parentID": "-1",
 "movedToBin": false,
```

```
"objectType": "tcdirectory",
"links": [
{
"rel": "self",
"href": "https://172.30.91.227:8443/umsapi/v3/directories/tcdirectories/15592"
},
{
"rel": "Parent",
"href": "root"
}
]
},
{
"id": "76863",
"name": "New Subdirectory",
"parentID": "76462",
"movedToBin": false,
"objectType": "tcdirectory",
"links": [
{
"rel": "self",
"href": "https://172.30.91.227:8443/umsapi/v3/directories/tcdirectories/76863"
},
{
"rel": "Parent",
"href": "https://172.30.91.227:8443/umsapi/v3/directories/tcdirectories/76462"
```

```
 }
 ]
 },
 {
 "id": "76462",
 "name": "New Directory",
 "parentID": "-1",
 "movedToBin": false,
 "objectType": "tcdirectory",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.227:8443/umsapi/v3/directories/tcdirectories/76462"
 },
 {
 "rel": "Parent",
 "href": "root"
 }
 ]
 }
]
```

Creating a Thin Client Directory

▶ Send an HTTP PUT request to the resource `/v3/directories/tcdirectories` to create a new thin client directory.

This request must contain *JSON* data in its body specifying a name for the new directory. The `parentID` is optional and defaults to the root thin client directory:

```
{
 "name":"My Directory",
 "parentID":"-1"
}
```

cURL accepts *JSON* data as the `--data` parameter. This request also specifies `Content-type` header to tell *IMI* that the data has the type `application/json`.

▶ Use it whenever you send *JSON* data:

```
curl \
--request PUT \
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
--header "Content-type: application/json" \
--data '{"name":"My Directory", "parentID":"-1"}' \
https://[server]:8443/umsapi/v3/directories/tcdirectories
```

Response

IMI replies with a success message and data about the newly created directory:

```
{
 "message": "Directory successfully inserted.",
 "id": "77118",
 "name": "My Directory",
 "parentID": "-1"
}
```

Updating a Thin Client Directory

▶ Send an HTTP PUT request to `/v3/directories/tcdirectories/[id]` to update the properties of a thin client directory. The directory ID is passed as a variable in the URL, a new directory name as *JSON* data in the request body. Do not forget the `Content-type` header.

```
curl \
--request PUT \
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
--header "Content-type: application/json" \
--data '{"name":"My Wonderful Directory"}' \
https://[server]:8443/umsapi/v3/directories/tcdirectories/77118
```

Response

▶ *IMI* replies with a success message:

```
200 OK
---
{
 "message": "Updated directory successfully."
}
```

Deleting a Thin Client Directory

▶ Send an HTTP DELETE request to `/v3/directories/tcdirectories/[id]` to delete a thin client directory. The ID of the directory is passed as a path variable in the URL.

```
curl \
--request DELETE \
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
https://[server]:8443/umsapi/v3/directories/tcdirectories/77118
```

Response

▶ *IMI* replies with a brief success message:

Code Example:

```
<code> 200 OK</code>
```

Code Example:

```
<code>---</code>
```

Code Example:

```
<code> {"message":"Deletion successful."} </code>
```

A thin client directory has to be empty to be removed. If you try to delete a non-empty directory, *IMI* will respond with an error message:

```
400 Bad Request
---
{
 "message": "Directory with ID <77118> is not empty. Remove content first.",
 "errorCode": "ITG-0A006",
 "time": "2016-04-06T13:29:59.362",
 "stackTrace": "[de.igel.rm.umsapi.server.services.thinclientdirectory.
```

```
[...]
}
```

Learn more about error codes and messages in the IMI API Reference (see page 90).

Further Operations

- Moving a Thin Client Directory (see page 34)
- Sending a Command to Thin Clients (see page 35)
- Debugging Requests (see page 37)

Moving a Thin Client Directory

▶ In order to move thin clients or directories send a PUT request to the target directory and append `?operation=move`. The request body must contain a list of API Objects in *JSON* format, representing the thin clients and directories to be moved. An API Object has an ID and a type.

```
curl \
--request PUT \
--header 'Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
--header "Content-type: application/json" \
--data '[ { "id":"77123", "type":"tcdirectory"},\
 { "id":"1234", "type":"tcdirectory" } ]' \
https://[server]:8443/umsapi/v3/directories/tcdirectories/15592?operation=move
```

Response

The following response contains an error message for one directory and a success message for the other:

```
200 OK
---
[
 {
 "id": "1234",
 "results": "does_not_exist"
 },
 {
 "id": "77123",
 "results": "successful"
 }
]
```

Sending a Command to Thin Clients

▶ Send a POST request to `/v3/thinclients?command=[command]` to send a command to one or more thin clients. A command can be one of the following:

- `reboot`
- `shutdown`
- `wakeup`
- `settings2tc`
- `tcreset2facdefs`

The request body contains a list of API Objects representing the thin clients addressed.

To send the `reboot` command to two thin clients, send the following request to *IMI*:

```
curl \
--request POST \
JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC' \
--header "Content-type: application/json" \
--data '[{"id":"27", "type":"tc"},{"id":"72014", "type":"tc"}]' \
https://[server]:8443/umsapi/v3/thinclients?command=reboot
```

Request

IMI replies with a *JSON* document containing a result for each thin client addressed:

```
{
 "CommandExecList": [
 {
 "execID": "ID-PM-MH-WIN7-UMS-54530-1456839861871-5-0",
 "id": "72014",
 "mac": "00E0C561EEED",
 "exectime": "1456845240566",
 "message": "OK",
```

```
 "state": "SUCCESS"
 },
 {
 "execID": "ID-PM-MH-WIN7-UMS-54530-1456839861871-5-0",
 "id": "27",
 "mac": "00E0C54DCB8E",
 "exectime": "1456845240560",
 "message": "OK",
 "state": "SUCCESS"
 }
 ]
}
```

Debugging Requests

Error Codes

To debug errors when communicating with IMI, observe

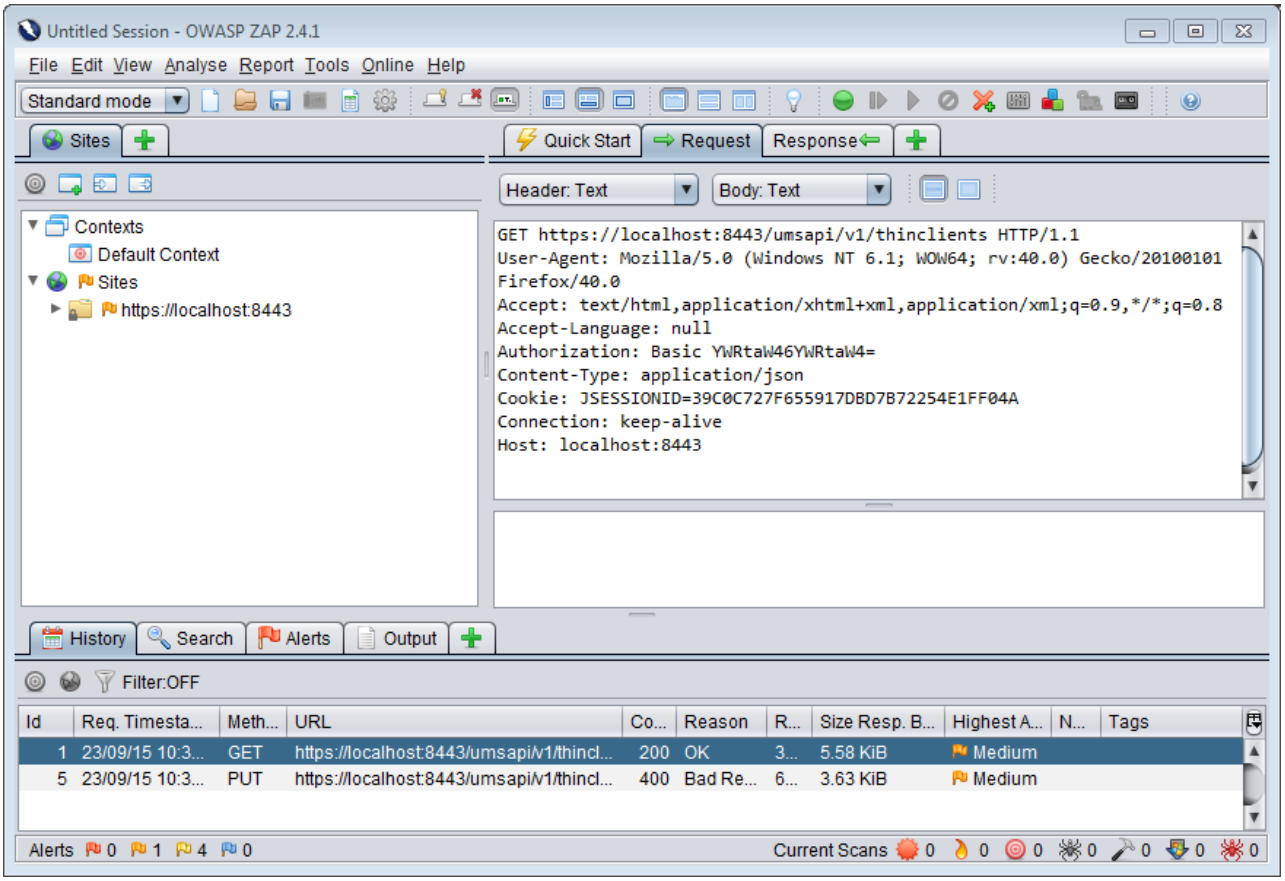
- the HTTP status in the response headers
- the messages and error codes in the response bodies.

HTTP-Proxy

You may use an HTTP proxy in order to see what requests your scripts or programs produce. A proxy acts as a man-in-the-middle between client and IMI, letting you view and optionally edit requests.

An Open Source program for this task is OWASP ZAP[3].

Use a proxy only for debugging your own HTTP requests. Viewing other users' network traffic is usually illegal.



3 https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

Powershell

- Introduction (see page 39)
- Command Reference (see page 43)

Introduction

IMI Cmdlets is a collection of *Windows PowerShell* commandlets for use with the *IGEL Management Interface (IMI)*. The commandlets are installed as binary `*.dll` files and can be used in *PowerShell* scripts. They simplify frequent actions such as rebooting a thin client or assigning a profile via *IMI*, hiding the implementation details of the underlying REST API.

> Microsoft .NET version 4.5 or greater is required for IMI Cmdlets 1.04.100 for TLSv1.2

- Installation (see page 40)
- Loading the IMI Cmdlets Snap-in (see page 41)
- Authentication (see page 42)

Installation

Prerequisites

- One of the following *Microsoft* operating systems:
  - *Windows 7*
  - *Windows Server 2008* or *2008 R2*
  - *Windows 8* or higher
  - *Windows Server 2012 or 2012 R2*
- *Windows Management Framework 4* (which already includes *PowerShell 4)*
- *Microsoft .Net Framework 4* or newer

If you are using UMS 5.08.100 or higher, you will need at least IMI PowerShell Cmdlets 1.04.100.
Lower versions of IMI PowerShell Cmdlets are not compatible with UMS 5.08.100 or higher, because this UMS version uses TLS 1.2.

Installing

1. Download the *IMI* Commandlets Installer from the download server[4].
2. Launch the installer.

   You need administration privileges in order to install IMI Commandlets.
3. Close all other applications and confirm that you have done so.
4. Review and accept the **License Agreement**.
5. Select the destination location for the installation or leave the default.
6. Optional: Select a Start Menu folder for the program shortcuts.
7. Read the summary and start the installation process.
   The installer will install the IMI Commandlets and some sample scripts. The following shortcuts will be placed on the desktop:
   - **Starter-Interface**: A simple interactive terminal interface to *IMI*, suitable for new users
   - **IGEL-Shell**: A *Windows* PowerShell session with *IMI* Commandlets already loaded

   Additionally, the *IMI* Commandlets Readme file is displayed.

4 https://www.igel.com/software-downloads/

Loading the IMI Cmdlets Snap-in

Before you can use IMI Cmdlets, you need to add the snap-in to your PowerShell session:

```
Add-PSSnapin *igel*
```

View all loaded IGEL Cmdlets:

```
Get-PSSnapin *igel*
```

Authentication

IMI uses HTTP Basic Authentication with a username-password combination. IMI Cmdlets wrap authentication in the **Get-IgelRMServerLogin** cmdlet.

Get-IgelRMServerLogin logs in to IMI.

Parameters

- **-Servername**
  Hostname of the of the UMS server with IMI
- **-Username**
  UMS user name
- **-ForceIMIVersion**
  Use a specific IMI version, `v1` or `v2.`
- **-IgnoreUntrustedCertificates**
  If this is set to `$true` the Cmdlets will ignore errors in TLS/SSL certificate validation.

The password is entered interactively.

Example Command Line

Log in and save the credentials for the session in `$l`:

```
$l = Get-IgelRMServerLogin `
-Servername localhost `
-Username SPIELWIESE `
-IgnoreUntrustedCertificates $true `
-ForceIMIVersion v2
```

Supply the credentials to subsequent commands using

`-Credentials $l`

Command Reference

- Thin Client (see page 44)
- Profile (see page 52)
- Master Profile (see page 59)
- Thin Client Directory (see page 66)
- Profile Directory (see page 74)
- Master Profile Directory (see page 80)
- Firmware (see page 86)
- Server Status (see page 88)

Thin Client

- Get-IgelTCInformation (see page 45)
- Start-IgelTCs (see page 47)
- Stop-IgelTC (see page 48)
- Restart-IgelTCs (see page 49)
- Move-IgelTCs (see page 50)
- Update-IgelTCSettings (see page 51)

Get-IgelTCInformation

Summary

Gets information on thin clients.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login.
- -**tcID**
  Numeric ID of the thin client. If omitted, information is retrieved on all thin clients, including those in the Recycling Bin.
- **-Details**
  Level of detail in the output, one of:
  - **full** Gets all data available from IMI, including the online status.
  - **inventory** Gets all data available from IMI, apart from the online status.
  - **online** Gets data in a short format, including the online status.

  If this parameter is omitted, only a short information format with the most important thin client properties is returned.

Example Command Line

```
Get-IgelTCInformation -Credentials $l
```

Example Output

```
mac : 000BCA050027
firmwareID : 45
lastIP : 172.30.91.237
id : 48335
name : UD2-D220
parentID : 76863
movedToBin : True
objectType : tc

mac : 005056934FDB
firmwareID : 28
```

```
lastIP : 172.30.91.9
id : 48366
name : westestx86
parentID : -1
movedToBin : False
objectType : tc

mac : 00E0C561EEED
firmwareID : 43
lastIP : 172.30.91.30
id : 72014
name : IGEL-8KJ2GQPL3N
parentID : 76863
movedToBin : False
objectType : tc
```

Start-IgelTCs

Summary

Wakes up thin clients.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**tcID / -tcIDs**
  Numeric IDs of the thin clients to wake up

Example Command Line

```
Start-IgelTCs -Credentials $l -tcIDs 27
```

Example Output

```
execID exectime message state
------ -------- ------- -----
ID-PM-MH-WIN7-UMS-49242-14... 1460647127627 OK SUCCESS
```

Stop-IgelTC

Summary

Shuts down thin clients.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**tcID / -tcIDs**
  Numeric IDs of the thin clients to shut down

Example Command Line

```
Stop-IgelTCs -Credentials $l -tcID 2010
```

Example Output

```
execID exectime message state
------ -------- ------- -----
ID-mhuber-50400-14605614057... 1460706641670 OK SUCCESS
```

Restart-IgelTCs

Summary

Reboots thin clients.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**tcIDs**
  Numeric IDs of the thin clients to reboot

Example Command Line

```
Restart-IgelTCs -Credentials $l -tcIDs 2010,1435
```

Example Output

```
execID exectime message state
------ -------- ------- -----
ID-mhuber-50400-14605614057... 1460991770313 OK SUCCESS
ID-mhuber-50400-14605614057... 1460991770313 OK SUCCESS
```

Move-IgelTCs

Summary

Moves a thin client to a thin client directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**tcIDs**
  Numeric ID of the thin client
- **-DirectoryID**
  Numeric ID of the target thin client directory

Example Command Line

```
Move-IgelTCs `
-Credentials $l `
-TCIDs 2010 `
-DirectoryID 3147
```

Example Output

```
id results
-- -------
2010 successful
```

Update-IgelTCSettings

Summary

Sends settings from UMS to the thin client.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**tcID**
  Numeric ID of the thin client

Example Command Line

```
Update-IgelTCSettings -Credentials $l -tcID 2010
```

Example Output

```
execID : ID-mhuber-50400-1460561405782-16-0
```

```
id : 2010
```

```
mac : 000BCA050027
```

```
exectime : 1460708102541
```

```
message : OK
```

```
state : SUCCESS
```

Profile

- Get-IgelTCProfile (see page 53)
- Rename-IgelTCProfile (see page 54)
- Get-IgelProfileAssignments (see page 55)
- Set-IgelProfileAssignment (see page 56)
- Remove-IgelProfileAssignment (see page 57)
- Remove-IgelTCProfile (see page 58)

Get-IgelTCProfile

Summary

Gets information about profiles.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile. If this is omitted, information is retrieved for all profiles

Example Command Line

```
Get-IgelTCProfile -Credentials $l -ProfileID 3150
```

Example Output

```
firmwareID : 3
isMasterProfile : False
overridesSessions : False
id : 3150
name : My Profile
parentID : -2
movedToBin : False
objectType : profile
```

Rename-IgelTCProfile

Summary

Renames a profile.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile
- **-newProfileName**
  New profile name as string

Example Command Line

```
Rename-IgelTCProfile `
-Credentials $l `
-ProfileID 3150 `
-newProfileName "Updated Profile"
```

Example Output

```
message
-------
Update successful
```

Get-IgelProfileAssignments

Summary

Gets profile assignments.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile
- **-ObjectType**
  Get assigments to a specific object type, one of:
  - **tc** thin client
  - **directory** thin client directory

Example Command Line

```
Get-IgelProfileAssignments `
-Credentials $l `
-ProfileID 3150 `
-ObjectType tc
```

Example Output

```
type id assignmentPosition
---- -- ------------------
tc 2010 0
```

Set-IgelProfileAssignment

Summary

Creates a profile or master profile assignment.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile or master profile
- **-ProfileType**
  Profile type, one of:
  - **profile** profile
  - **masterprofile** master profile
- -**TargetID**
  Numeric ID of the target object which the profile or master profile will be assigned to
- **-TargetType**
  Assignments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Set-IgelProfileAssignment `
-Credentials $l `
-ProfileID 3150 `
-ProfileType profile `
-TargetID 2010 `
-TargetType tc
```

Example Output

```
message
-------
1 asssignments successfully assigned
```

Remove-IgelProfileAssignment

Summary

Deletes a profile or master profile assignment.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile or master profile
- **-ProfileType**
  Profile type, one of:
  - **profile** profile
  - **masterprofile** master profile
- -**TargetID**
  Numeric ID of the target object which the profile or master profile is assigned to
- **-TargetType**
  Assignments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Remove-IgelProfileAssignment `
-Credentials $l `
-ProfileID 3151 `
-ProfileType profile `
-TargetType tcdirectory `
-TargetID 3201
```

Example Output

```
message
-------
deleted profile assignment
```

Remove-IgelTCProfile

Summary

Deletes a profile.

> This does not move the profile to the Recycling Bin, but simply deletes it.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile

Example Command Line

```
Remove-IgelProfile -Credentials $l -ProfileID 3150
```

Example Output

```
message
-------
Deleted profile with id 3150
```

Master Profile

- Get-IgelTCMasterprofile (see page 60)
- Rename-IgelTCMasterprofile (see page 61)
- Get-IgelMasterprofileAssignments (see page 62)
- Set-IgelProfileAssignment (see page 63)
- Remove-IgelProfileAssignment (see page 64)
- Remove-IgelMasterprofile (see page 65)

Get-IgelTCMasterprofile

Summary

Gets information about master profiles.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**MasterprofileID**
  Numeric ID of the master profile. If this is omitted, information is retrieved for all master profiles

Example Command Line

```
Get-IgelTCMasterprofile -Credentials $l -MasterprofileID 3152
```

Example Output

```
firmwareID : 3
isMasterProfile : True
overridesSessions : False
id : 3152
name : My Master Profile
parentID : -14
movedToBin : False
objectType : masterprofile
```

Rename-IgelTCMasterprofile

Summary

Renames a master profile.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**MasterprofileID**
  Numeric ID of the master profile
- **-newMasterProfileName**
  New master profile name as string

Example Command Line

```
Rename-IgelTCMasterprofile `
-Credentials $l `
-MasterprofileID 3152 `
-newMasterProfileName "Renamed Master Profile"
```

Example Output

```
message
-------
Update successful
```

Get-IgelMasterprofileAssignments

Summary

Gets master profile assignments.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**MasterprofileID**
  Numeric ID of the master profile
- **-ObjectType**
  Get assigments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Get-IgelMasterprofileAssignments `
-Credentials $l `
-MasterprofileID 3152 `
-ObjectType tc
```

Example Output

```
type id assignmentPosition
---- -- ------------------
tc 2010 0
```

Set-IgelProfileAssignment

Summary

Creates a profile or master profile assignment.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile or master profile
- **-ProfileType**
  Profile type, one of:
  - **profile** profile
  - **masterprofile** master profile
- -**TargetID**
  Numeric ID of the target object which the profile or master profile will be assigned to
- **-TargetType**
  Assignments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Set-IgelProfileAssignment `
-Credentials $l `
-ProfileID 3150 `
-ProfileType profile `
-TargetID 2010 `
-TargetType tc
```

Example Output

```
message
-------
1 asssignments successfully assigned
```

Remove-IgelProfileAssignment

Summary

Deletes a profile or master profile assignment.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile or master profile
- **-ProfileType**
  Profile type, one of:
  - **profile** profile
  - **masterprofile** master profile
- -**TargetID**
  Numeric ID of the target object which the profile or master profile is assigned to
- **-TargetType**
  Assignments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Remove-IgelProfileAssignment `
-Credentials $l `
-ProfileID 3151 `
-ProfileType profile `
-TargetType tcdirectory `
-TargetID 3201
```

Example Output

```
message
-------
deleted profile assignment
```

Remove-IgelMasterprofile

Summary

Deletes a master profile.

> This does not move the master profile to the Recycling Bin, but simply deletes it.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**MasterprofileID**
  Numeric ID of the master profile

Example Command Line

```
Remove-IgelMasterprofile -Credentials $l -MasterprofileID 3152
```

Example Output

```
message
-------
Deleted profile with id 3152
```

Thin Client Directory

- Get-IgelTCDirectories (see page 67)
- Rename-IgelTCDirectory (see page 69)
- Set-IgelTCDirectory (see page 70)
- Move-IgelTCDirectory (see page 71)
- Set-IgelProfileAssignment (see page 72)
- Remove-IgelTCDirectory (see page 73)

Get-IgelTCDirectories

Summary

Gets information on thin client directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**DirectoryID**
  Numeric ID of the thin client directory. If omitted, information is retrieved on all thin client directories, including those in the Recycling Bin
- **-details**
  Amount of detail in the output:
  - **$true** The IDs of `DirectoryChildren` will be shown, which can be thin clients or thin client directories.

Example Command Line

```
Get-IgelTCDirectories -Credentials $l -details $true
```

Example Output

```
id : 3201
name : B Directory
parentID : -1
movedToBin : False
objectType : tcdirectory
DirectoryChildren : {3202, 1435}

id : 3202
name : Subdirectory
parentID : 3201
movedToBin : False
objectType : tcdirectory
```

```
DirectoryChildren : {}

id : 3147
name : A Directory
parentID : -1
movedToBin : False
objectType : tcdirectory
DirectoryChildren : {2010}
```

Rename-IgelTCDirectory

Summary

Renames a thin client directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**directoryID**
  Numeric ID of the thin client directory
- **-newDirectoryName**
  New thin client directory name as string

Example Command Line

```
Rename-IgelTCDirectory `
-Credentials $l `
-directoryID 3206 `
-newDirectoryName "Renamed TC Directory"
```

Example Output

```
message
-------
Updated directory successfully.
```

Set-IgelTCDirectory

Summary

Creates a new thin client directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryName**
  Name for the thin client directory
- -**directoryPosition**
  Parent directory for the new thin client directory. If omitted, it defaults to `-1`, which is the UMS directory **Thin Clients**.

Example Command Line

```
Set-IgelTCDirectory `
-Credentials $l `
-directoryName "Brand New Directory" `
-directoryPosition 3201
```

Example Output

```
message id name parentID
------- -- ---- --------
Directory successfully inse... 3206 Brand New Directory 3201
```

Move-IgelTCDirectory

Summary

Moves one or more thin client directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryIDs**
  Numerical IDs of the thin client directories to be moved
- -**targetDirectory**
  The target directory to move the thin client directories into

Example Command Line

```
Move-IgelTCDirectory `
-Credentials $l `
-directoryIDs 3206 `
-targetDirectory
```

Example Output

```
id results
-- -------
3206 successful
```

Set-IgelProfileAssignment

Summary

Creates a profile or master profile assignment.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**ProfileID**
  Numeric ID of the profile or master profile
- **-ProfileType**
  Profile type, one of:
  - **profile** profile
  - **masterprofile** master profile
- -**TargetID**
  Numeric ID of the target object which the profile or master profile will be assigned to
- **-TargetType**
  Assignments to a specific object type, one of:
  - **tc** thin client
  - **tcdirectory** thin client directory

Example Command Line

```
Set-IgelProfileAssignment `
-Credentials $l `
-ProfileID 3150 `
-ProfileType profile `
-TargetID 2010 `
-TargetType tc
```

Example Output

```
message
-------
1 asssignments successfully assigned
```

Remove-IgelTCDirectory

Summary

Deletes a thin client directory.

> You can only delete empty thin client directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryID**
  Numerical ID of the thin client directory to be deleted

Example Command Line

```
Remove-IgelTCDirectory -Credentials $l -directoryID 3202
```

Example Output

```
message
-------
Deletion successful.
```

Profile Directory

- Get-IgelProfileDirectory (see page 75)
- Rename-IgelTCProfileDirectory (see page 76)
- Move-IgelTCProfileDirectory (see page 77)
- Set-IgelProfileDirectory (see page 78)
- Remove-IgelProfileDirectory (see page 79)

Get-IgelProfileDirectory

Summary

Gets information on profile directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**DirectoryID**
  Numeric ID of the profile directory. If omitted, information is retrieved on all profile directories, including those in the Recycling Bin.

Example Command Line

```
Get-IgelProfileDirectory -Credentials $l -ProfileDirectoryID 3221
```

Example Output

```
id : 3221
```

```
name : Profiles 2
```

```
parentID : -2
```

```
movedToBin : False
```

```
objectType : profiledirectory
```

Rename-IgelTCProfileDirectory

Summary

Renames a profile directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**directoryID**
  Numeric ID of the profile directory
- **-newDirectoryName**
  New profile directory name as string

Example Command Line

```
Rename-IGELTCProfileDirectory `
-Credentials $l `
-directoryID 3220 `
-newDirectoryName "My Renamed Directory"
```

Example Output

```
message
-------
Updated directory successfully.
```

Move-IgelTCProfileDirectory

Summary

Moves one or more profile directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryIDs**
  Numerical IDs of the profile directories to be moved
- -**targetDirectory**
  The target directory to move the profiles directories into

Example Command Line

```
Move-IgelProfileDirectory `
-Credentials $l `
-directoryIDs 4024 `
-targetDirectory 3220
```

Example Output

```
id results
-- -------
4024 successful
```

Set-IgelProfileDirectory

Summary

Creates a new profile directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryName**
  Name for the profile directory
- -**directoryPosition**
  Parent directory for the new profile directory. If omitted, it defaults to `-2`, which is the UMS directory **Profiles**.

Example Command Line

```
Set-IgelProfileDirectory `
-Credentials $l `
-directoryName "A New Directory" `
-directoryPosition -2
```

Example Output

```
message id name parentID
------- -- ---- --------
Directory successfully inse... 4030 A New Directory -2
```

Remove-IgelProfileDirectory

Summary

Deletes a profile directory.

> You can only delete empty thin client directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryID**
  Numerical ID of the profile directory to be deleted

Example Command Line

```
Remove-IgelProfileDirectory -Credentials $l -ProfileDirectoryID 4024
```

Example Output

```
message
-------
Deletion successful.
```

Master Profile Directory

- Get-IgelMasterprofileDirectory (see page 81)
- Rename-IgelTCMasterprofileDirectory (see page 82)
- Move-IgelMasterprofileDirectory (see page 83)
- Set-IgelMasterprofileDirectory (see page 84)
- Remove-IgelMasterprofileDirectory (see page 85)

Get-IgelMasterprofileDirectory

Summary

Gets information on master profile directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**DirectoryID**
  Numeric ID of the master profile directory. If omitted, information is retrieved on all master profile directories, including those in the Recycling Bin

Example Command Line

```
Get-IgelMasterprofileDirectory -Credentials $l -MasterprofileDirectoryID 4034
```

Example Output

```
id : 4034
```

```
name : New Master Profiles
```

```
parentID : -14
```

```
movedToBin : False
```

```
objectType : masterprofiledirectory
```

Rename-IgelTCMasterprofileDirectory

Summary

Renames a master profile directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**directoryID**
  Numeric ID of the master profile directory
- **-newDirectoryName**
  New master profile directory name as string

Example Command Line

```
Rename-IGELTCMasterProfileDirectory `
-Credentials $l `
-directoryID 4038 `
-newDirectoryName "Renamed Master Profiles"
```

Example Output

```
message
-------
Updated directory successfully.
```

Move-IgelMasterprofileDirectory

Summary

Moves one or more master profile directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryIDs**
  Numerical IDs of the master profile directories to be moved
- -**targetDirectory**
  The target directory to move the master profiles directories into

Example Command Line

```
Move-IgelMasterprofileDirectory `
-Credentials $l `
-directoryIDs 4033 `
-targetDirectory 4034
```

Example Output

```
id results
-- -------
4033 successful
```

Set-IgelMasterprofileDirectory

Summary

Creates a new master profile directory.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryName**
  Name for the master profile directory
- -**directoryPosition**
  Parent directory for the new master profile directory. If omitted, it defaults to `-14`, which is the UMS directory **Master Profiles.**

Example Command Line

```
Set-IgelMasterprofileDirectory -Credentials $l -directoryName "Brand New Directory"
```

Example Output

```
message id name parentID
------- -- ---- --------
Directory successfully inse... 4038 Brand New Directory -14
```

Remove-IgelMasterprofileDirectory

Summary

Deletes a master profile directory.

> You can only delete empty directories.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- **-directoryID**
  Numerical ID of the master profile directory to be deleted

Example Command Line

```
Remove-IgelMasterprofileDirectory -Credentials $l -MasterprofileDirectoryID 403 3
```

Example Output

```
message
-------
Deletion successful.
```

Firmware

- Get-IgelFirmware (see page 87)

Get-IgelFirmware

Summary

Gets information on the firmwares registered with UMS.

Parameters

- -**Credentials**
  Credential data, usually passed via a variable which it has been saved to at login
- -**firmwareID**
  Numeric ID of the firmware. If omitted, information is retrieved on all firmwares.

Example Command Line

```
Get-IgelFirmware -Credentials $l -firmwareID 3
```

Example Output

```
id product version firmwareType
```

```
-- ------- ------- ------------
```

```
3 IGEL Universal Desktop LX 5.09.100.01 LX
```

Server Status

- Get-IgelRMServerStatus (see page 89)

Get-IgelRMServerStatus

Summary

Gets information on the UMS Server and IMI. No authentication is necessary for this.

Parameters

- **-Servername**
  Hostname of the UMS server with IMI
- **-IgnoreUntrustedCertificates**
  If this is set to `$true` the Cmdlets will ignore errors in TLS/SSL certificate validation.

Example Command Line

```
Get-IgelRMServerStatus -Servername 172.30.91.158 -IgnoreUntrustedCertificates $true
```

Example Output

```
buildNumber : 26712
rmGuiServerVersion : 5.02.100.rc8
activeMQVersion : 5.6.0
derbyVersion : 10.12.1.1
serverUUID : a400f9da-333d-49d8-bb8d-2238543a4643
server : mhuber:8443
```

IMI API V1 Reference

This document describes the resources and methods made available by the *IGEL Management Interface API Version 1* in a reference format with one entry per method.

_____

- Prerequisites (see page 91)
- Client Applications and Libraries (see page 92)
- Authentication (see page 94)
- Resources (see page 98)
- Error Codes (see page 147)

Prerequisites

IGEL Universal Management Suite (UMS)

UMS 5.01.100

Networking

In order to use *IGEL Management Interface* you need to be able to reach the API host via the network and connect to its API port, TCP 8443 by default.

The base URL is

- `https://[server]:8443/umsapi/`

The resources for IMI version 1 are available at:

- `https://[server]:8443/umsapi/v1/`

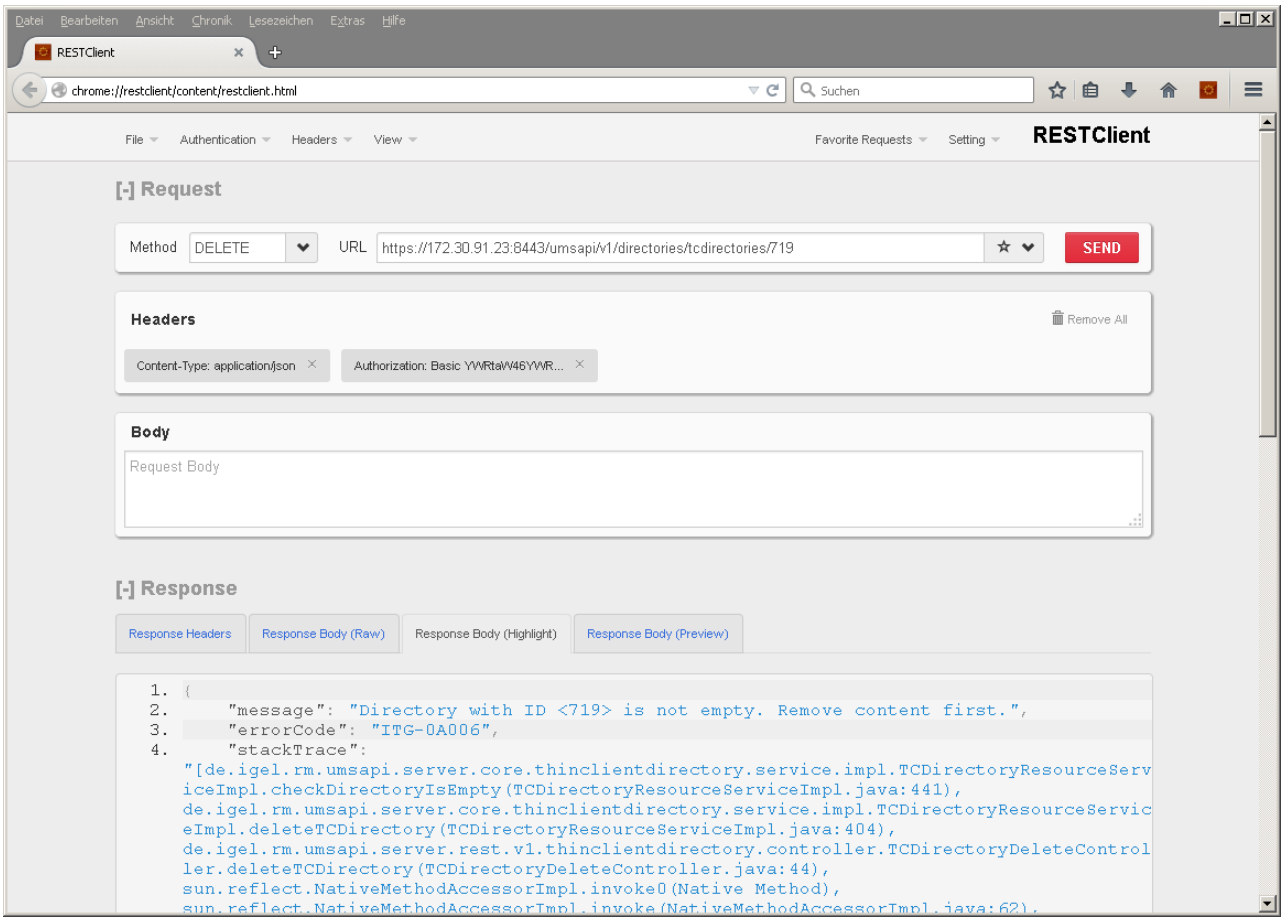
> *IMI* uses HTTPS to ensure the integrity and confidentiality of the network traffic. It is good practice to use a valid server certificate with a verifiable signature. Most clients and libraries can be configured to work with self-signed or invalid certificates, but that should not be done in a production environment.

Client Applications and Libraries

Clients

The easiest way to try out the *IGEL Management Interface* is either

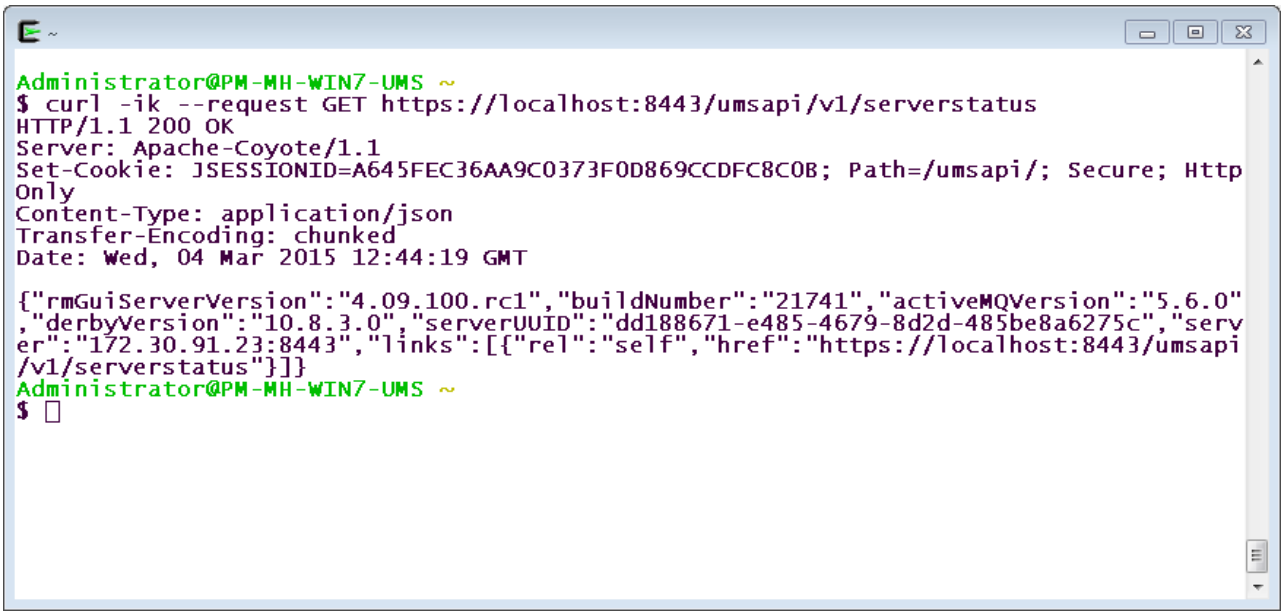
- with RESTClient[5], an add-on for *Mozilla Firefox:*



or

- with cURL[6], a commandline network client:

5 http://restclient.net/
6 http://curl.haxx.se/



Both are licensed as Open Source software and are available free of charge.

This guide uses *cURL* for examples, as the commandline makes all parameters visible in plain text. This need not stop you from using *RESTClient* if you are more comfortable with it. You can easily translate the commandline parameters into the fields of the *RESTClient* GUI.

Programming

Most programming languages provide an HTTP and an SSL/TLS module, either in their standard library or as an extension, and a *JSON* library (for the API data format) as well.

Authentication

You must be authenticated in order to use *IGEL Management Interface,* otherwise the server will return the HTTP status 401 "Unauthorized". Only querying the server status is allowed without authentication.

The login mechanism uses HTTP Basic Authentication (RFC 2617).

_____

- POST /v1/login (see page 95)
- POST /v1/logout (see page 97)

POST /v1/login

Summary

Authenticates the client in regard to the *IMI* server.

Resource URL

`/v1/login`

Example Request

```
curl \
--request POST \
--user 'admin:W00t' \
https://[server]:8443/umsapi/v1/login
```

Response Body

name	type	description
`message`	String	String containing the JSESSIONID

Example Response

```
200 OK
Set-Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC; Path=/umsapi/; Secure; HttpOnly
---
{"message":"JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC"}
```

⚠ You can maintain the session by sending the JSESSIONID in the Cookie header with every subsequent request. Some clients will do this automatically for you, e.g. the *RESTClient Firefox* add-on.

```
Cookie: JSESSIONID=3FB2F3F6A089FE9029DFD6DAFEF146DC
```

POST /v1/logout

Summary

Logs the client out from the *IMI* session.

Resource URL

`/v1/logout`

Example Request

```
curl \
--request POST \
--header 'Cookie: JSESSIONID=11048CDA77DE2B45BE1562C8EED67858' \
https://[server]:8443/umsapi/v1/logout
```

Example Response

`200 OK`

```
{"message":"Logout successful"}
```

Resources

- Thin Client (see page 99)
- Thin Client Directory (see page 121)
- Firmware (see page 136)
- Server Status (see page 141)
- Entry Point (see page 143)
- Further Resources (see page 145)

Thin Client

Summary

Describes a thin client.

Properties

Name	Type	Mandatory	Description
`id`	String	no	Thin client ID
`mac`	String (12)	yes	MAC address
`firmwareID`	String	yes	Firmware ID
`parentID`	String	no	ID of parent directory
`name`	String	no	Human-readable name
`movedToBin`	Boolean	no	In Recycle Bin?
`networkName`	String	no	Network name
`site`	String	no	Site
`department`	String	no	Department
`lastIP`	String	no	Last known IP
`costCenter`	String	no	Cost center
`comment`	String	no	Comment
`assetID`	String	no	Asset ID
`inserviceDate`	String	no	Placed in service on
`serialNumber`	String	no	Serial Number
`productId`	String	no	Product ID

`umsStructuralTag`	String	no	Structure Tag
`cpuSpeed`	Integer	no	CPU speed (MHz)
`cpuType`	String	no	CPU type
`deviceType`	String	no	Device type
`deviceSerialNumber`	String	no	Device serial number
`osType`	String	no	Operating system type
`flashSize`	Integer	no	Flash size (MB)
`memorySize`	Integer	no	Memory size (MB)
`networkSpeed`	Integer	no	Network speed
`duplexMode`	String	no	Duplex mode
`graphicsChipset0`	String	no	First Graphics chipset
`graphicsMemorySize0`	Integer	no	First Graphics memory (MB)
`graphicsChipset1`	String	no	Second Graphics chipset
`graphicsMemorySize1`	Integer	no	Second Graphics memory (MB)
`monitorVendor1`	String	no	First Monitor vendor
`monitorModel1`	String	no	First Monitor model
`monitorSerialnumber1`	String	no	First Monitor serial number
`monitorSize1`	Number	no	First Monitor size
`monitorNativeResolution1`	String	no	First Monitor native resolution
`monitor1YearOfManufacture`	String	no	First Monitor year of manufacture

`monitor1WeekOfManufacture`	String	no	First Monitor week of manufacture
`monitorVendor2`	String	no	Second Monitor vendor
`monitorModel2`	String	no	Second Monitor model
`monitorSerialnumber2`	String	no	Second Monitor serial number
`monitorSize2`	Number	no	Second Monitor size
`monitorNativeResolution2`	String	no	Second Monitor native resolution
`monitor2YearOfManufacture`	String	no	Second Monitor year of manufacture
`monitor2WeekOfManufacture`	String	no	Second Monitor week of manufacture
`biosVendor`	String	no	BIOS vendor
`biosVersion`	String	no	BIOS version
`biosDate`	String	no	BIOS date
`totalUsagetime`	String	no	Total usage time
`totalUptime`	String	no	Total uptime
`lastBoottime`	String	no	Last boot time

GET /v1/thinclients

Summary

Gets information on all thin clients registered with the UMS instance.

> ⚠ This method will also list thin clients that are located in the **Recycle Bin** (`"movedToBin":"true"`), but you will not be able to call methods on those clients.

Resource URL

`/v1/thinclients` /

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/thinclients
```

Response Type

Returns a list of thin client (see page 99) resources.

Example Response

```
{
 "TCResources": [
 {
 "id": "2731",
 "mac": "00E0C53C3881",
 "firmwareID": "4",
 "parentID": "-1",
 "name": "reception thin client",
```

```
 "movedToBin": false,
 "networkName": "IGEL-00E0C53C3881",
 [...]
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/v1/thinclients/2731"
 },
 {
 "rel": "Parent",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/v1/directories/
tcdirectories/-1"
 },
 {
 "rel": "Firmware",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/v1/firmwares/4"
 },
 {
 "rel": "Commands",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/v1/thinclients/
command"
 }
 ]
 },
 {
 "id": "2739",
 "mac": "008064AD82FB",
```

```
 "firmwareID": "2",
 "parentID": "-1",
 [...]
 }
 }
}
```

GET /v1/thinclients/[id]

Summary

Gets information on the specified thin client.

Resource URL

`/v1/thinclients/[id]`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/thinclients/[id]
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client ID

Response Type

Returns a thin client (see page 99) resource.

Example Response

```
{
 "id": "2731",
 "mac": "00E0C53C3881",
 "firmwareID": "4",
 "parentID": "-1",
 "name": "reception thin client",
 "movedToBin": false,
 "networkName": "IGEL-00E0C53C3881",
```

```
 "site": "main campus",
 "lastIP": "172.30.91.117",
 "productId": "UD5-720 LX",
 "cpuSpeed": 0,
 "cpuType": "",
 "deviceType": "",
 "deviceSerialNumber": "",
 "osType": "",
 "flashSize": 0,
 "memorySize": 0,
 "networkSpeed": 0,
 "graphicsChipset0": "",
 "graphicsMemorySize0": 0,
 "graphicsChipset1": "",
 "graphicsMemorySize1": 0,
 "monitorVendor1": "",
 [...]
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2731/v1/thinclients/2731"
 },
 {
 "rel": "Parent",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2731/v1/directories/tcdirectories/-1"
```

```
 },
 {
 "rel": "Firmware",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2731/v1/firmwares/4"
 },
 {
 "rel": "Commands",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2731/v1/thinclients/command"
 }
 ]
}
```

GET /v1/thinclients/[id]/status

Summary

Gets the online status of the specified thin client.

Resource URL

`/v1/thinclients/[id]/status`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/thinclients/2731/status
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client ID

Response Body

Name	Type	Description
`id`	String	thin client ID
`online`	Boolean	thin client online status
`lastBoottime`	String	last boot time

Example Response

```
Status: 200 OK
---
{
 "id": "2731",
```

```
 "online": false,
 "lastBoottime": "26.02.2015 15:53"
}
```

PUT /v1/thinclients

Summary

Creates a new thin client resource.

Resource URL

`/v1/thinclients`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"mac":"00E0C53C3881", \
"firmwareID":"2"}' \
https://[server]:8443/umsapi/v1/thinclients/
```

Request Body

Name	Type	Mandatory	Description
`mac`	String (12)	yes	thin client MAC adress
`firmwareID`	String	yes	firmware ID
`name`	String	no	thin client name
`parentID`	String	no	ID of parent directory
`site`	String	no	thin client site
`department`	String	no	department
`costCenter`	String	no	cost center
`lastIP`	String	no	last known IP of thin client

`comment`	String	no	comment field
`assetID`	String	no	asset ID
`inserviceDate`	String	no	
`serialNumber`	String	no	serial number of thin client

Response Body

Name	Type	Description
`message`	String	success or error message
`id`	String	thin client ID
`parentID`	String	ID of parent directory
`name`	String	thin client name

Example Response

```
Status: 200 OK
---
{
 "message": "Thin client successfully inserted.",
 "id": "7735",
 "name": "My Name",
 "parentID": "-1"
}
```

PUT /v1/thinclients/[id]

Summary

Updates properties of an existing thin client.

Resource URL

`/v1/thinclients/[id]`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"name":"reception thin client", \
"site":"main campus"}' \
https://[server]:8443/umsapi/v1/thinclients/123
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client ID

Request Body

Name	Type	Mandatory	Description
`name`	String	no	thin client name
`site`	String	no	site of operation
`department`	String	no	department
`costCenter`	String	no	cost center
`lastIP`	String	no	last known IP of thin client

`comment`	String	no	comment field
`assetID`	String	no	asset ID
`inserviceDate`	String	no	
`serialNumber`	String	no	serial number of thin client

Response Type

Returns a message.

Example Response

```
{
 "message": "Update successful"
}
```

DELETE /v1/thinclients/[id]

Summary

Deletes a thin client.

> This does not move the thin client into the **Recycle Bin** but simply deletes it.

Resource URL

`/v1/thinclients/[id]`

Example Request

```
curl \
--request DELETE \
https://[server]:8443/umsapi/v1/thinclients/2704
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client ID

Response Type

Returns a message.

Example Response

```
200 OK
---
{
 "CommandExecList": [
```

```
 {
 "execID": "ID-PM-MH-WIN7-UMS-63885-1424682219085-5-0",
 "mac": "008064AD82FB",
 "message": "OK",
 "exectime": 1424698605821,
 "state": "SUCCESS"
 }
 ]
}
```

DELETE /v1/thinclients/deletetcoffline/[id]

Summary

Deletes a thin client even if it is offline.

> This does not move the thin client into the **Recycle Bin** but simply deletes it.

Resource URL

`/v1/thinclients/deletetcoffline/[id]`

Example Request

```
curl \
--request DELETE \
https://[server]:8443/umsapi/v1/thinclients/deletetcoffline/2704
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client ID

Response Type

Returns a message.

Example Response

```
200 OK
---
 {
 "message": "Offline deletion successful"
 }
```

PUT /v1/thinclients/moveto/[id]

Summary

Moves one or more thin clients to the specified directory.

Resource URL

`/v1/thinclients/moveto/[id]`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"idList":["34", "49"]}' \
https://[server]:8443/umsapi/v1/thinclients/moveto/89
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	ID of the target directory

Request Body

Name	Type	Mandatory	Description
`idList`	IDListResource (see page 146)	`yes`	List of thin client IDs

Response Body

Name	Type	Description
`id`	String	directory ID

`result`	String	one of "successful", "target_is_parent", "does_not_exist", "is_in_bin"

Example Response

```
Status: 200 OK
---
{
 "MoveResponseList": [
 {
 "id": "34",
 "results": "successful"
 },
 {
 "id": "49",
 "results": "does_not_exist"
 }
 ]
}
```

POST /v1/thinclients/command

Summary

Sends a command to a thin client.

Resource URL

`/v1/thinclients/command`

Example Request

```
curl \
--request POST \
--header "Content-type: application/json" \
--data '{ "command":"reboot", \
"list": {"idList":["2984","2731"]}}' \
https://[server]:8443/umsapi/v1/thinclients/command
```

Request Body

Name	Type	Mandatory	Description
`command`	CommandsResource	yes	one of "reboot", "shutdown" or "wakeup"
`list`	IDListResource (see page 146)	`yes`	List of thin client IDs

Example Response

```
200 OK
---
Example Response
```

```
{
 "CommandExecList": [
 {
 "execID": "ID-PM-MH-WIN7-UMS-63885-1424682219085-9-0",
 "mac": "008064AD82FB",
 "message": "OK",
 "exectime": 1424765031363,
 "state": "SUCCESS"
 },
 {
 "execID": "ID-PM-MH-WIN7-UMS-63885-1424682219085-9-1",
 "mac": "00E0C53C3881",
 "message": "Connection timed out: connect",
 "exectime": 1424765031363,
 "state": "FAILED"
 }
 ]
}
```

Thin Client Directory

Summary

Describes a thin client directory.

Properties

Name	Type	Mandatory	Description
`id`	String	no	Thin client ID
`name`	String	yes	human-readable name
`parentID`	String	no	ID of parent directory
`movedToBin`	Boolean	no	true if in Recycle Bin
`controlledObjectType`	String	no	
`DirectoryChildrenResource`	directoryChildrenResource	can be empty	

GET /v1/directories/tcdirectories

Summary

Lists all thin client directories recursively.

Note

This method will also list thin client directories that are located in the Recycle Bin (`"movedToBin":"true"`).

Resource URL

`/v1/directories/tcdirectories`

Example Request

```
curl \
--request GET\
https://[server]:8443/umsapi/v1/directories/tcdirectories
```

Response Type

Returns a list of thin client directory (see page 121) resources.

Example Response

```
200 OK ---
{
 "DirectoryResources": [
 {
 "id": "75",
 "name": "Directory in Bin",
 "parentID": "-1",
 "movedToBin": true,
 "controlledObjectType": "tcdirectory",
```

```
"DirectoryChildrenResource": {
"DirectoryChildrenResource": [],
"links": []
},
"links": [
{
"rel": "self",
"href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories/75"
},
{
"rel": "root",
"href": "tcdirectory"
}
]
},
{
"id": "93",
"name": "Another Directory",
"parentID": "-1",
"movedToBin": false,
"controlledObjectType": "tcdirectory",
"DirectoryChildrenResource": {
"DirectoryChildrenResource": [
{
"controlledObjectType": "tc",
"id": "2984"
```

```
 }
 ],
 "links": [
 {
 "rel": "tc",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2984"
 }
 ]
 },
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories/93"
 },
 {
 "rel": "root",
 "href": "tcdirectory"
 }
 ]
 },
 {
 "id": "719",
 "name": "Yet Another Directory",
 "parentID": "-1",
 "movedToBin": false,
 "controlledObjectType": "tcdirectory",
```

```
"DirectoryChildrenResource": {
"DirectoryChildrenResource": [
{
"controlledObjectType": "tc",
"id": "2731"
}
],
"links": [
{
"rel": "tc",
"href": "https://172.30.91.23:8443/umsapi/v1/thinclients/2731"
}
]
},
"links": [
{
"rel": "self",
"href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories/719"
},
{
"rel": "root",
"href": "tcdirectory"
}
]
}
]
```

```
}
```

GET /v1/directories/tcdirectories/[id]

Summary

Gets information on a thin client directory.

Resource URL

`/v1/directories/tcdirectories/[id]`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/directories/tcdirectories/123
```

Response Type

Returns a thin client directory (see page 121) resource.

Response Body

Name	Type	Description
`id`	String	Thin client directory ID
`name`	String	Human-readable name
`parentID`	String	Parent directory ID
`movedToBin`	Boolean	True if directory is in Recycle Bin
`controlledObjectType`	String	`tc` or `tcdirectory`
`DirectoryChildrenResource`	directoryChildrenResource	

Example Response


```
{
 "id": "73",
 "name": "A Directory",
 "parentID": "-1",
 "movedToBin": false,
 "controlledObjectType": "tcdirectory",
 "DirectoryChildrenResource": {
 "DirectoryChildrenResource": [
 {
 "controlledObjectType": "tcdirectory",
 "id": "93"
 }
 ],
 "links": [
 {
 "rel": "tcdirectory",
 "href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories/93"
 }
 ]
 },
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories/73"
 },
```

```
 {
 "rel": "root",
 "href": "tcdirectory"
 }
 ]
}
```

PUT /v1/directories/tcdirectories

Summary

Creates a new thin client directory.

Resource URL

`/v1/directories/tcdirectories`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"name":"My Directory", "parentID":"-1"}' \
https://[server]:8443/umsapi/v1/directories/tcdirectories
```

Request Body

Name	Type	Mandatory	Description
`name`	String	`yes`	human-readable name
`parentID`	String	no	parent directory ID, default: root directory (-1)

Response Body

Name	Type	Description
`message`	String	success message
`id`	String	id of newly created directory
`name`	String	human-readable name
`parentID`	String	parent directory ID

Example Response

```
200 OK
---
{
 "message": "Directory successfully inserted",
 "id": "3035",
 "name": "My Directory",
 "parentID": "-1"
}
```

PUT /v1/directories/tcdirectories/[id]

Summary

Updates properties of an existing thin client directory.

Resource URL

`/v1/directories/tcdirectories[id]`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"name":"New Directory Name"}' \
https://[server]:8443/umsapi/v1/directories/tcdirectories/123
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	directory ID

Request Body

Name	Type	Mandatory	Description
`name`	String	no	new directory name

Example Response

```
200 OK
---
{
 "message": "Updated directory successfully"
}
```

PUT /v1/directories/tcdirectories/moveto/[id]

Summary

Moves one or more thin client directories into another directory.

Resource URL

`/v1/directories/tcdirectories/moveto/[id]`

Example Request

```
curl \
--request PUT \
--header "Content-type: application/json" \
--data '{"idList":["93","709"]}' \
https://[server]:8443/umsapi/v1/directories/tcdirectories/moveto/123
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	target directory ID

Request Body

Name	Type	Mandatory	Description
`idList`	idListResource (see page 146)	yes	list of directory IDs

Response Body

Name	Type	Description
`id`	String	directory ID

`result`	String	one of "successful", "target_is_child", "target_is_parent", "is_target", "does_not_exist"

Example Response

```
200 OK
---
{
 "MoveResponseList": [
 {
 "id": "93",
 "results": "successful"
 },
 {
 "id": "709",
 "results": "does_not_exist"
 }
 ]
}
```

DELETE /v1/directories/tcdirectories/[id]

Summary

Deletes a thin client directory. The directory has to be empty.

> This does not move the directory into the **Recycle Bin** but simply deletes it.

Resource URL

`/v1/directories/tcdirectories/[id]`

Example Request

```
curl \
--request DELETE \
https://[server]:8443/umsapi/v1/directories/tcdirectories/3035
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	thin client directory ID

Example Response

```
200 OK
---
{
 "message": "Deletion successful"
}
```

Firmware

Summary

Describes a firmware resource.

Properties

Name	Type	Mandatory	Description
`id`	String	yes	numeric ID
`product`	String	no	Product name, e.g. IGEL Universal Desktop OS 2
`version`	String	no	Version number, e.g. 5.05.100.rc5.01
`firmwareType`	String	no	Firmware family, e.g. LX

GET /v1/firmwares

Summary

Lists all firmwares available on the UMS instance.

Resource URL

`/v1/firmwares`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/firmwares
```

Response Type

Returns a list of Firmware (see page 136) resources.

Example Response

```
Status: 200 OK
---
{
  "FwResource": [
    {
      "id": "2",
      "product": "IGEL Universal Desktop OS 2",
 "version": "5.05.100.rc5.01",
 "firmwareType": "LX",
 "links": [
 {
 "rel": "self",
```

```
 "href": "https://172.30.91.23:8443/umsapi/v1/firmwares/v1/firmwares/2"
 }
 ]
 },
 {
 "id": "4",
 "product": "IGEL Universal Desktop LX",
 "version": "4.10.200.01",
 "firmwareType": "LX",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/firmwares/v1/firmwares/4"
 }
 ]
 }
 ]
}
```

GET /v1/firmwares/[id]

Summary

Gets information on a firmware resource.

Resource URL

`/v1/firmwares/[id]`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/firmwares/2
```

Request Path Variables

Name	Type	Mandatory	Description
`id`	String	yes	ID of the firmware

Response Type

Returns a Firmware (see page 136) Resource.

Response Body

Name	Type	Description
`id`	String	firmware ID
`product`	String	product name
`version`	String	version
`firmwareType`	String	firmware type

Example Response

```
Status: 200 OK
```

```
---
{
 "id": "2",
 "product": "IGEL Universal Desktop OS 2",
 "version": "5.05.100.rc5.01",
 "firmwareType": "LX",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/firmwares/2/v1/firmwares/2"
 }
 ]
}
```

Server Status

Summary

Gets information about the UMS server instance.

Resource URL

`/v1/serverstatus`

Example Request

```
curl \
--request GET\
https://[server]:8443/umsapi/v1/serverstatus
```

Example Response

```
200 OK
---
{
 "rmGuiServerVersion": "4.09.100",
 "buildNumber": "21678",
 "activeMQVersion": "5.6.0",
 "derbyVersion": "10.8.3.0",
 "serverUUID": "dd188671-e485-4679-8d2d-485be8a6275c",
 "server": "172.30.91.23:8443",
 "links": [
 {
 "rel": "self",
 "href": "https://172.30.91.23:8443/umsapi/v1/serverstatus"
```

```
  }
  ]
}
```

Entry Point

Summary

Provides a list of links to the resources available via the *IGEL Management Interface*. This makes the API discoverable by following the links.

Resource URL

`/v1/`

Example Request

```
curl \
--request GET \
https://[server]:8443/umsapi/v1/
```

Example Response

```
200 OK
{
 "links": [
 {
 "rel": "Login",
 "href": "https://172.30.91.23:8443/umsapi/v1/login"
 },
 {
 "rel": "Thin Clients",
 "href": "https://172.30.91.23:8443/umsapi/v1/thinclients"
 },
 {
 "rel": "Thin Client Directories",
```

```
 "href": "https://172.30.91.23:8443/umsapi/v1/directories/tcdirectories"
 },
 {
 "rel": "Firmwares",
 "href": "https://172.30.91.23:8443/umsapi/v1/firmwares"
 }
 ]
}
```

Further Resources

- iDListResource (see page 146)

iDListResource

An array containing thin client IDs.

Example data

```
["34", "49", "662"]
```

Error Codes

The *IGEL Management Interface* uses HTTP status codes as well as API error messages in the *JSON* body to tell you about errors that have occurred.

_____

- HTTP Status Codes (see page 148)
- API Error Messages (see page 149)

HTTP Status Codes

Client-Side Errors

HTTP Status	Title	Description
200	OK	The request was successful. However, it is a good idea to read the API message.
400	Bad Request	The request does not match the API, e.g. it uses the wrong HTTP method
401	Unauthorized	The client has not logged in or has sent the wrong credentials.
404	Not Found	The endpoint does not exist, it may be misspelled.
415	Unsupported Media Type	The body content, e.g. JSON, does not match the Content-Type header or is not well-formed.

Server-Side Errors

HTTP Status	Title	Description
500	Internal Server Error	The server has encountered an error, check the server logfiles `catalina.log` and `stderr`

API Error Messages

The API server will send a message in the response body, should an error occur:

```
{
 "message": "Request method 'GET' not supported",
 "errorCode": "ITG-0A011",
 "stackTrace": "[org.springframework.web.servlet.mvc.m ...
 [...]
}
```

Error Code	Message
Client Errors	
ITG-0A001	Thin client with ID [id] not found.
ITG-0A002	No target directory with ID [id] found.
ITG-0A003	[command] is an invalid command.
ITG-0A004	Invalid request body
ITG-0A005	"Could not read JSON: [...]"
ITG-0A006	Directory with ID [id] is not empty. Remove content first.
ITG-0A007	No MAC defined.
ITG-0A008	[...] The statement was aborted because it would have caused a duplicate key value [...]
ITG-0A009	No firmware defined.
ITG-0A010	Number format error
ITG-0A011	Request method [method] not supported.
ITG-0A012	IDList is empty.
ITG-0A013	No valid property in body
ITG-0A014	MAC must be 12 characters long.
ITG-0A015	Firmware with ID [id] does not exist.
ITG-0A016	Parent with ID [id] does not exist.
ITG-0A017	Target directory is in bin. Please revert it first.

ITG-0A017	Thin client is in bin. Please revert it first.
Authentication Errors	
ITG-0B001	No authentication header found in request.
ITG-0B002	Invalid login - credentials.
ITG-0B004	No session ID found.
ITG-0B004	No session found for requested session ID [id].
ITG-0B005	Access denied.
ITG-0B006	No valid IMI APII version.
ITG-0B007	No user found in session with sessionID
Licensing Errors	
ITG-0C001	License required
Server Errors	
ITG-FF001	Internal error
ITG-FF002	Database error
ITG-FF003	Cache error
ITG-FF004	Server not available