TeamDrive Agent Documentation

Release 5.2.2

TeamDrive Systems GmbH

Jul 07, 2025

CHAPTER

ONE

COPYRIGHT NOTICE

TeamDrive Systems GmbH
https://www.teamdrive.com
Max-Brauer-Allee 50
22765 Hamburg, Germany
Email: info@teamdrive.com

CHAPTER

TWO

INTRODUCTION

This manual will guide you through the installation of your TeamDrive Agent. This document is intended for system administrators and users, who need to install and configure a TeamDrive Agent.

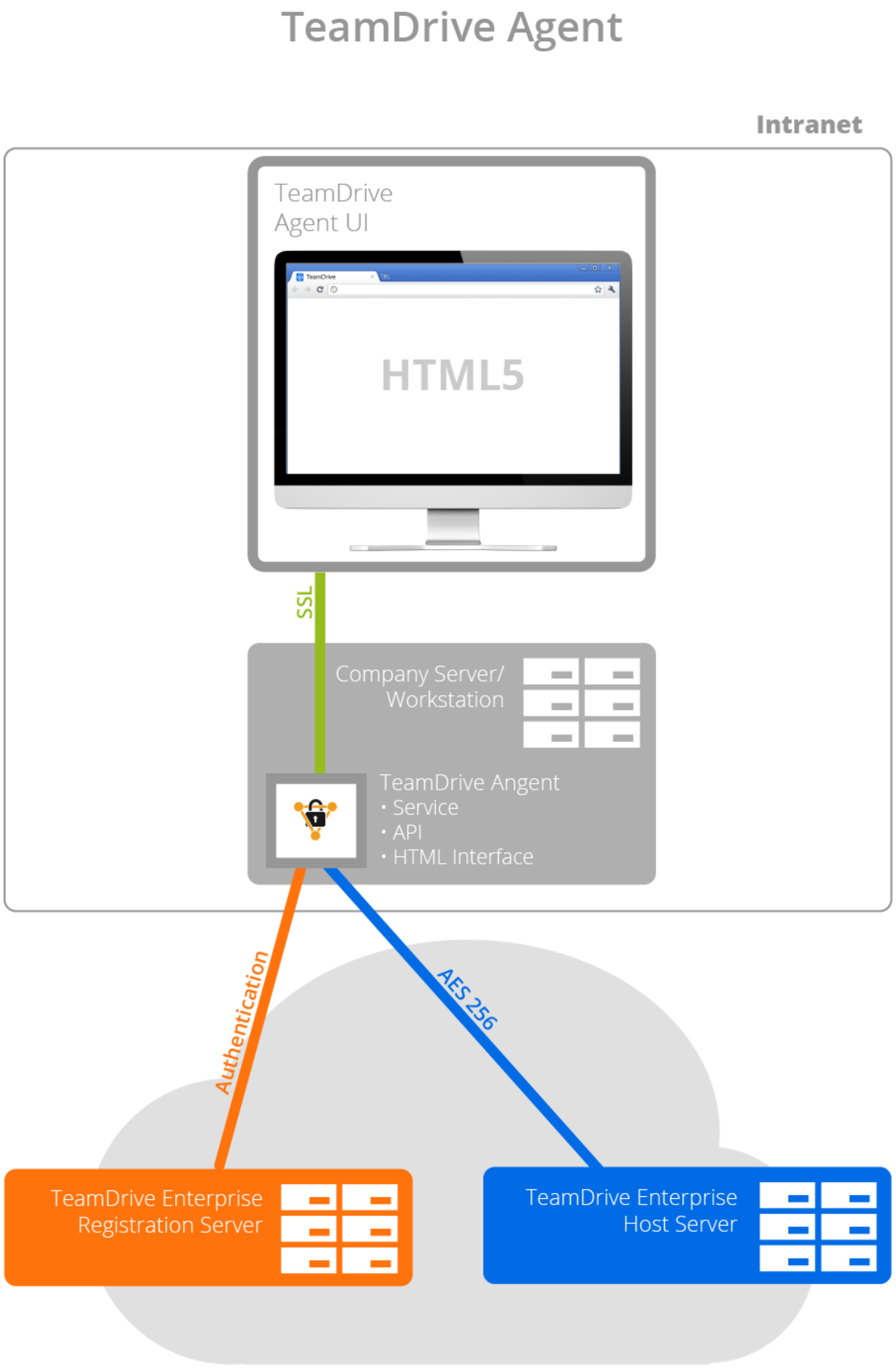
This Installation Guide outlines the deployment of a stand alone TeamDrive Agent, which is not part of a Web Portal.

2.1 Overview

The TeamDrive Agent is a full featured TeamDrive Client without a graphical user interface. An Agent provides:

- Synchronization of Spaces with the Cloud.
- Synchronization of Spaces with a local file system.
- Synchronization of Spaces with a remote file system.
- Management GUI using a Web App.
- API access using JSON calls.

This figure shows an overview of a TeamDrive Agent installation:


TeamDrive Agent
Intranet
TeamDrive
Agent UI
HTML5
SSL
Company Server/
Workstation
TeamDrive Angent
• Service
• API
• HTML Interface
Authentication
AES 256
TeamDrive Enterprise
Registration Server
TeamDrive Enterprise
Host Server

Typically, a TeamDrive Agent is installed on a server or network attached storage and communicates with the TeamDrive Cloud.

CHAPTER

THREE

WINDOWS INSTALLATION

3.1 Requirements

The *TeamDrive Agent* allows a TeamDrive Windows client to be installed and run as a Windows service. The *TeamDrive Agent* can be run without a GUI and allows data to be synchronized in the background, even when a Windows user is not currently logged in.

The support of older Windows versions has been dropped. The minimum supported Windows version is **Windows 7**, **Windows 10** or **Windows Server 2012**, **Windows Server 2016**, **Windows Server 2019**.

In order to install the *TeamDrive Agent*, you will need at least 50 MB of free disk space, a valid email address and Internet access. In the end, the total amount of disk space used will depend on the size of your created and/or joined Spaces.

If it is your first time installing and using TeamDrive you will need to create a TeamDrive account before installing the *TeamDrive Agent*. TeamDrive cannot be used without an account.

3.2 Required Skills

When installing the *TeamDrive Agent*, we assume that you have basic knowledge of:

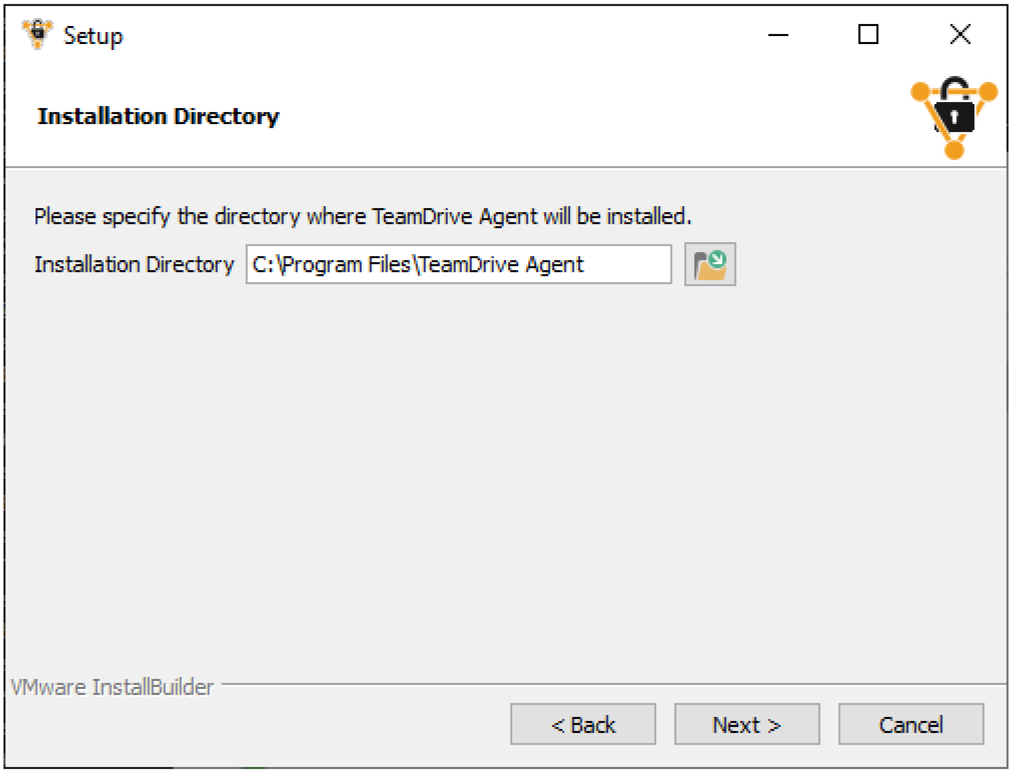
- Starting and stopping of Windows Services
- Configuration of a firewall

3.3 Installation

After you have successfully downloaded the *TeamDrive Agent*, double-click the installation file to start the installation.

Note: In order to install the *TeamDrive Agent* on a Windows terminal server, you need to put the server into a special installation mode by executing *Change user /install* at a command prompt.

1. To install the *TeamDrive Agent*, please specify an installation directory as seen in the following screen shot. We recommend not changing the installation directory to another location, as using the default directory will help with future version upgrades.

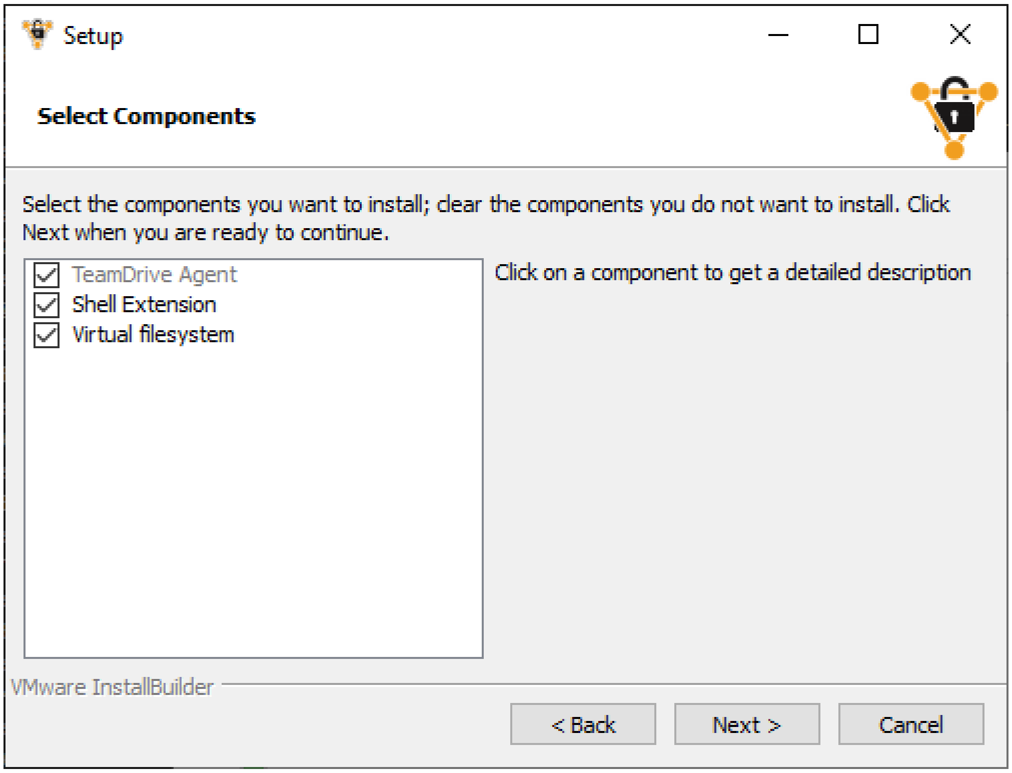

Setup

Installation Directory

Please specify the directory where TeamDrive Agent will be installed.

Installation Directory: C:\Program Files\TeamDrive Agent

2. Afterwards, you can choose to install a Windows Explorer integration which provides some context menu entries and overlay icons that show the current synchronization status of Spaces, folders and files:


Setup

Select Components

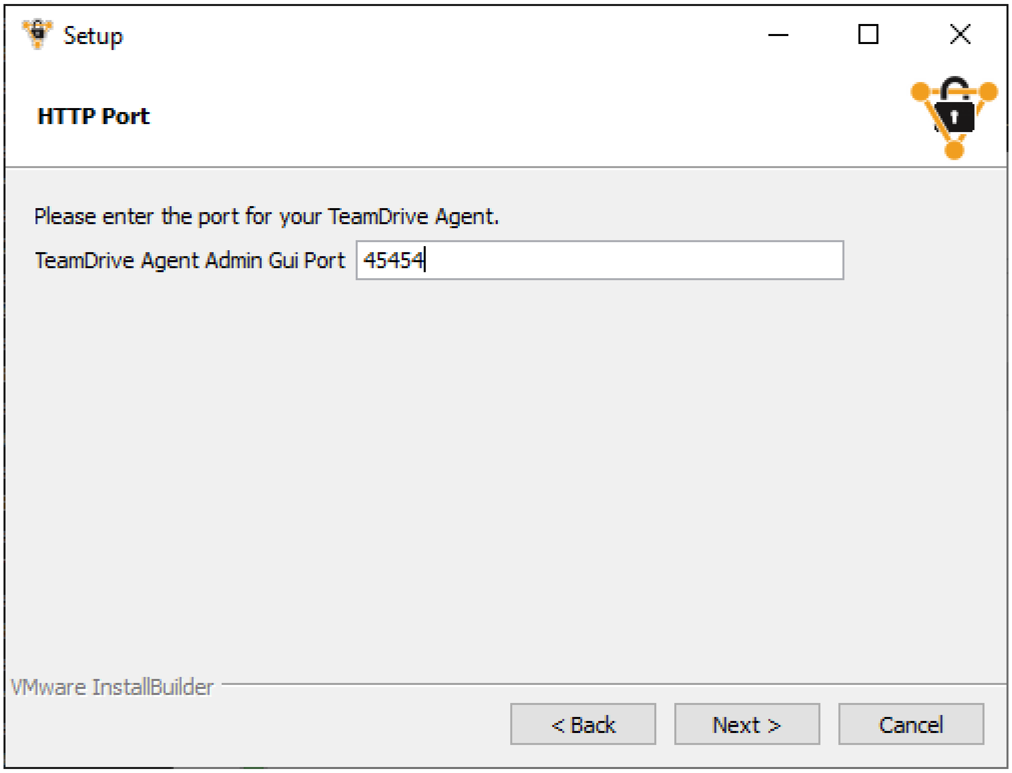
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

☑ TeamDrive Agent
☑ Shell Extension
☑ Virtual filesystem

Click on a component to get a detailed description

3. In order to administer the *TeamDrive Agent*, you will need to specify a TCP port. This port will be used to provide an administration interface using your web browser. You can choose another port number, but further

examples in this document will use the port *45454*:



Setup

HTTP Port

Please enter the port for your TeamDrive Agent.

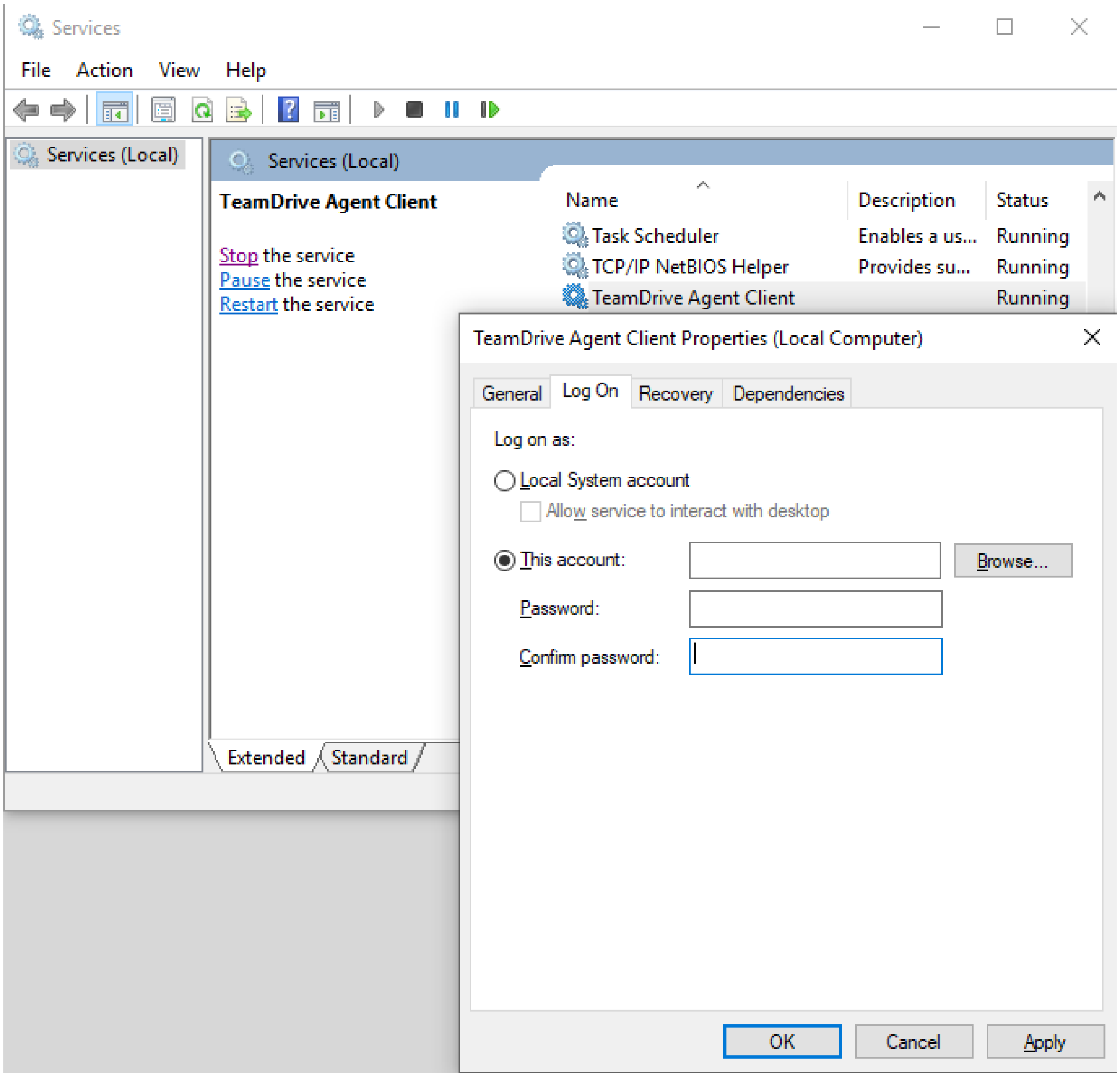
TeamDrive Agent Admin Gui Port 45454

4. During the installation, the *TeamDrive Agent* will be registered as a Windows service under the name "*TeamDrive Agent Client*". Please be aware that the Windows user logged in at the time of the *TeamDrive Agent Client's* installation is automatically associated with this Windows service. Only this Windows user will be able to access the settings file of the *TeamDrive Agent*.

3.4 Post-Installation Tasks

To be able to synchronize files with the *TeamDrive Agent*, some initial steps are required.

1. Open the Windows services overview (e.g. on Windows 8 "Control Panel" → "Administrative Tools" → "Services").
2. Please choose the user account under which the "*TeamDrive Agent Client*" will run. Notice: The default LocalSystem account can not be used because of the limited privileges:



Services (Local)

TeamDrive Agent Client

Stop the service
Pause the service
Restart the service

Name	Description	Status
Task Scheduler	Enables a us...	Running
TCP/IP NetBIOS Helper	Provides su...	Running
TeamDrive Agent Client		Running

TeamDrive Agent Client Properties (Local Computer)

Log on as:

○ Local System account

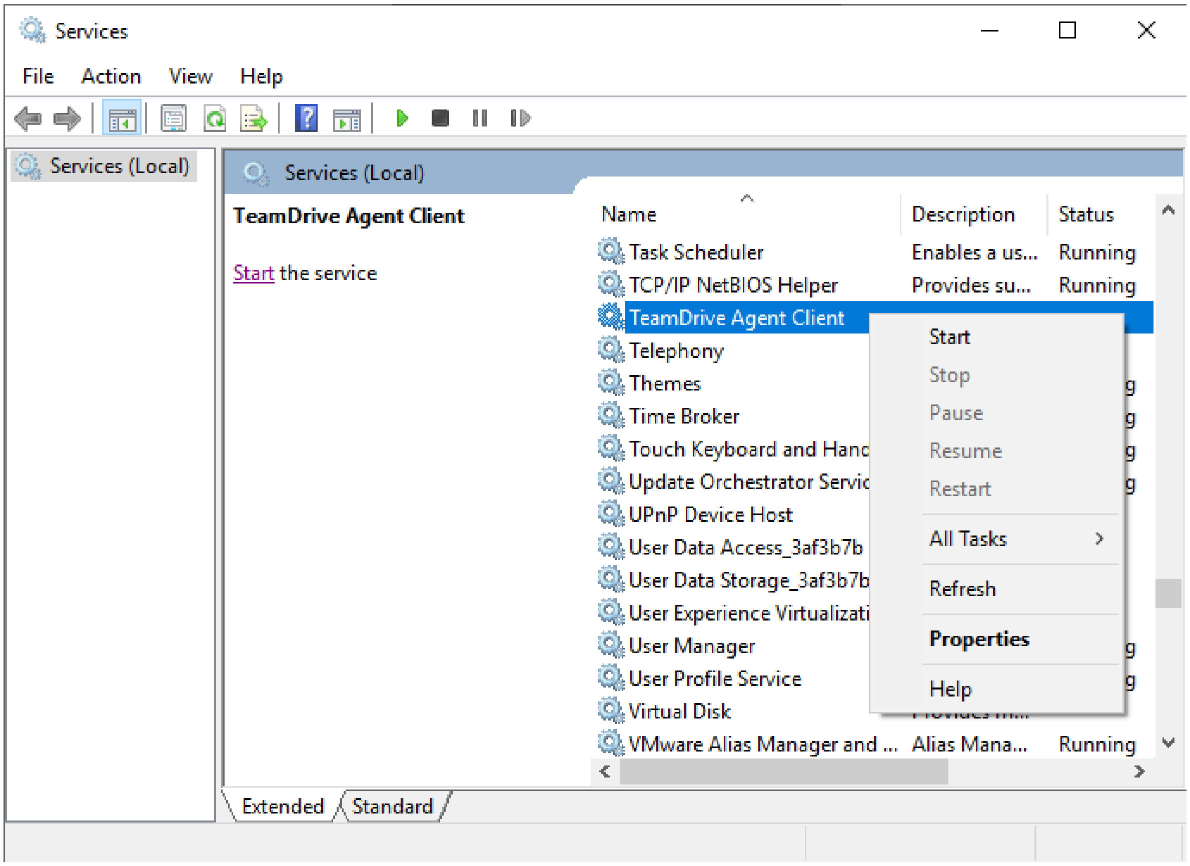
□ Allow service to interact with desktop

● This account:

Password:

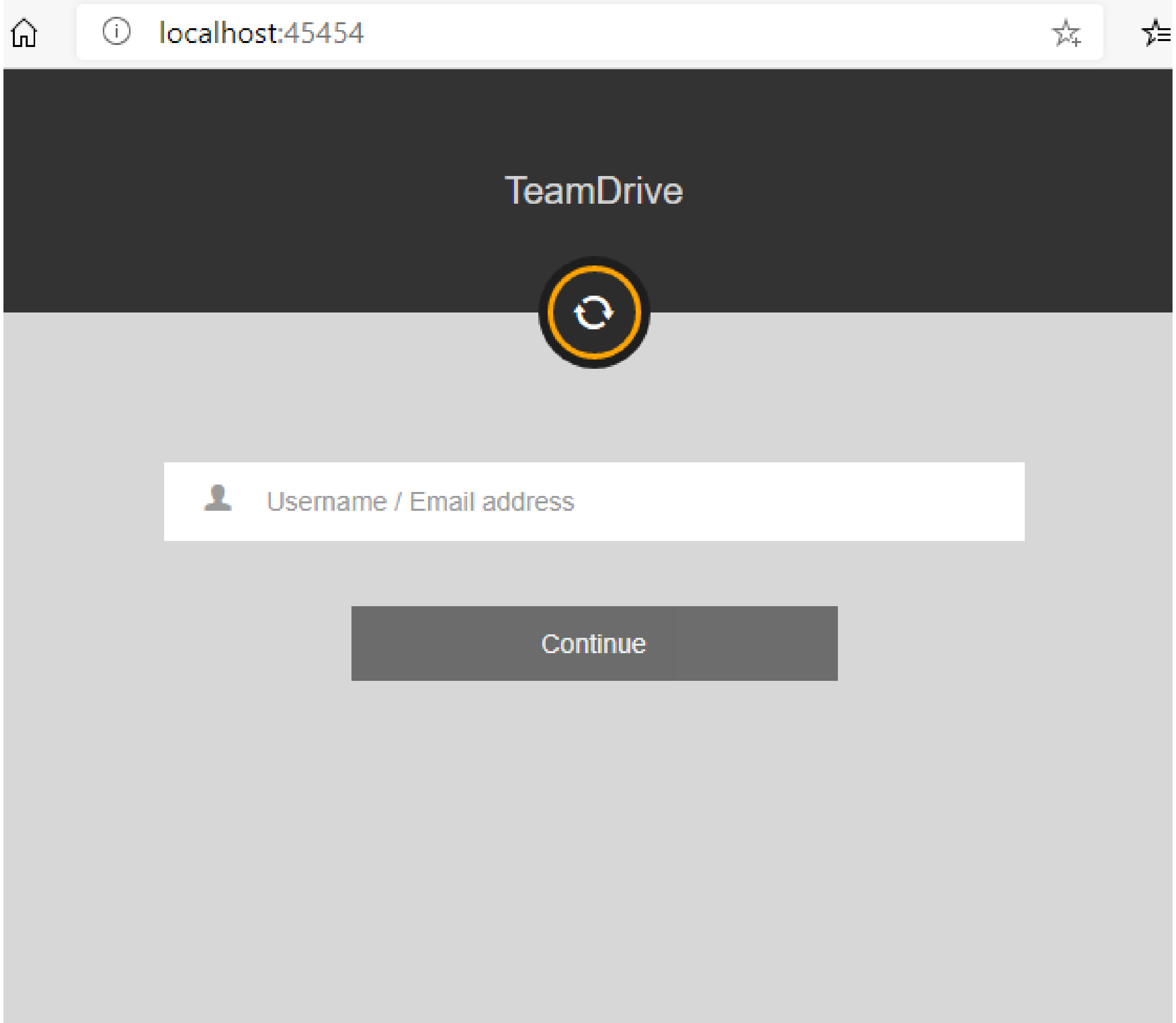
Confirm password:

3. Please start the Windows service "*TeamDrive Agent Client*":



Name	Description	Status
Task Scheduler	Enables a us...	Running
TCP/IP NetBIOS Helper	Provides su...	Running
TeamDrive Agent Client		
Telephony		
Themes		g
Time Broker		g
Touch Keyboard and Hanc		g
Update Orchestrator Servic		g
UPnP Device Host		
User Data Access_3af3b7b		
User Data Storage_3af3b7b		
User Experience Virtualizati		
User Manager		g
User Profile Service		g
Virtual Disk	Provides m...	
VMware Alias Manager and ...	Alias Mana...	Running

4. Open your browser and go to http://localhost:45454. Please replace the port with the one that was used during installation:



You can now activate Spaces in the administration interface. Your TeamDrive Spaces will now be synchronized in the background.

3.5 Changing the Windows User Account

If at some point you wish to change which Windows user you would like to use to manage the *TeamDrive Agent*, you simply need to edit the configuration file “teamdrive.ini”.

You can navigate to the file using “%ProgramData%” → “TeamDrive” → “teamdrive.ini” or simply *C:\ProgramData\TeamDrive\teamdrive.ini*. Open the file with a text editor, locate the entry “teamdrive-home-agent=” and replace the old Windows user account with the new desired Windows user account in the given file path. Here is an example of a typical *teamdrive.ini* that was created by the installer:

```
[Settings]
teamdrive-home-agent=C:/Users/teamdrive/AppData/Roaming/TeamDrive
http-api-port-agent=45454
shell-extension-user-agent=teamdrive
```

Please also look at the documentation of *teamdrived --teamdrive-home* (page 21), *teamdrived --http-api-port* (page 23) and *Platform specific Settings* (page 24).

The setting *shell-extension-user-agent* must be set to the same Windows username as in the *teamdrive-home* setting.

Afterwards, please continue with *Post-Installation Tasks* (page 7).

3.6 Accessing Network Shares

In order to access Spaces on network shares, the Windows account needs to be changed to a user, who has rights to access network shares. It is recommended to use the Windows account used to install the *TeamDrive Agent*.

Note: the *TeamDrive Agent* running as a Windows Services may not be able to access mapped network dries. It is recommended to use a *UNC* path directly as a Space root.

3.7 Uninstalling the TeamDrive Agent

To completely remove the TeamDrive Agent and all files from you System, follow these Steps:

1. Stop the *TeamDrive Agent Client* Windows Service
2. Execute the *uninstall.exe* executable and follow the wizard.
3. You may now delete the `teamdrived --teamdrive-home` (page 21) folder containing the application data.
4. If you have used a different *TeamDrive Data* path, please also delete this folder. See `teamdrived --data-path` (page 21) for details.
5. You can now delete the Spaces path as configured in `teamdrived --spaces-path` (page 21). Keep in mind that this folder may contain your Space key backups. Without the keys, you may no longer be able to access your Spaces.
6. Open a Administrator console and execute this command to also remove the Windows Service:

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>sc delete "TeamDrive Agent Client"
[SC] DeleteService SUCCESS
```

CHAPTER FOUR

LINUX INSTALLATION

At the moment, you can either use the 64 bit RPM package or the 64 bit .tar.gz file.

4.1 Requirements

The Library Dependencies are moderate and should be provided by all recent linux distributions:

Library	Version
GLibc	2.9
GLibc++	3.4.11
libz	

Under Linux, TeamDrive is using the inotify [1] API of the Linux kernel to be notified about file modifications. Under some Linux distributions, there is a rather small limit regarding the number of files that can be watched. You can find out the current watch handle limit by executing `cat /proc/sys/fs/inotify/max_user_watches` in a terminal. If this number is lower than the number of files in your Spaces, please increase this value by adding for example `fs.inotify.max_user_watches=100000` to your */etc/sysctl.conf* file. Please reboot your system afterwards.

4.2 Installation on RPM based Distributions

If you are on a RPM based distribution, please install the TeamDrive Agent by installing the rpm file.

The TeamDrive Agent is hosted on a dedicated yum repository. This makes the installation and applying of future updates very easy — you can simply run yum update to keep your TeamDrive Agent up to date.

To enable the repository, you need to download the td-agent.repo file and place it into the directory */etc/yum.repos.d/*, e.g. by using *wget*:

```
[root@agent ~]# wget -O "/etc/yum.repos.d/td-agent.repo" \
"http://repo.teamdrive.net/td-agent.repo"
```

This will enable the “TeamDrive Agent Version 4” repository, which you can check by running *yum repolist* afterwards:

```
[root@agent ~]# yum repolist
Loaded plugins: security
repo id            repo name                                  status
td-agent-4         TeamDrive Agent Version 4                  4
base               CentOS-6 - Base                            6.367
extras             CentOS-6 - Extras                          14
```

[1] http://man7.org/linux/man-pages/man7/inotify.7.html

```
updates             CentOS-6 - Updates                        1.094
repolist: 7.477
```

To install the TeamDrive Agent, install the following package via yum from the “TeamDrive Agent” repository:

```
[root@agent ~]# yum install teamdrive-Agent
```

Please continue with *Starting and Stopping* (page 13).

4.3 Installation on other Distributions

To install the TeamDrive Agent on a non-RPM based distribution, please download the Agent manually for example by using wget:

```
wget http://s3download.teamdrive.net/4.<version>/TMDR/linux-x86_64/TeamDrive_Agent_4.
↪<version>_x86_64.tar.gz
```

Please replace <*version*> with the latest release. Now, please extract the *TeamDrive Agent*:

```
tar -xvzf TeamDrive_Agent_4.<version>_x86_64.tar.gz
```

This will extract the TeamDrive Agent into a subdirectory *TeamDrive*.

Note: The *tar.gz* file does not register a *teamdrive-Agent* command on your system.

4.4 Starting and Stopping

Before starting the TeamDrive Agent for the first time, please also read *Initial Set-Up* (page 13).

Please start TeamDrive by using the shortcut in your desktop menu or bar. Alternatively, you can execute the global command “teamdrive-Agent” or ”./TeamDrive” from the installation folder. TeamDrive requires some system settings that are not available by executing the teamdrived.bin executable directly.

Note: Running TeamDrive as root is not recommended. TeamDrive runs perfectly without privileges.

To stop the TeamDrive Agent, please execute:

```
/opt/teamdrive/TeamDriveApi.py '<agent-host>:<port>' quit False
```

Replace <*agent-host*> with the host where your TeamDrive Agent is installed and the port with the one you specified.

4.5 Initial Set-Up

To administrate your TeamDrive Agent, you need to enable the HTTP API by setting the `teamdrived --http-api-port` (page 23) option either as a command line argument or as a setting in the file `~/.teamdrive/teamdrive.settings`:

```
[Settings]
http-api-port=45454
http-api-type=tcp
```

Then start the TeamDrive Agent, open your favorite browser and go to *http://<agent-host>:45454*. Replace *<agent-host>* with the host where your TeamDrive Agent is installed and the port with the one you specified. Log-in to your TeamDrive account now.

To start the agent after a reboot on a CentOS 7 system, please follow the systemd reference guide:

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/sect-Managing_Services_with_systemd-Unit_Files.html#sect-Managing_Services_with_systemd-Unit_File_Create

Create a file `/etc/systemd/system/teamdrive.service` with the following content (replace the `User=teamdrive` with your useraccount and also change `Environment` and `ExecStart` to your setup):

```
[Unit]
Description=teamdrive Service
After=network.target

[Service]
Type=simple
User=teamdrive
Environment=LD_LIBRARY_PATH=/home/teamdrive/teamdrive
ExecStart=/home/teamdrive/teamdrive/teamdrived.bin
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

Notify systemd that a new `teamdrive.service` file exists and start the service by executing the following command as root:

```
systemctl daemon-reload
systemctl start teamdrive.service
```

4.6 Activate SSL communication for the Agent web-interface

There are 2 possibilities to active SSL communication:

1. Use the build-in SSL communication with the following settings in the `teamdrive.settings` file (you can use your own port instead of the default ssl port 443):

```
[Settings]
http-api-port=443
http-api-type=ssl
http-api-certificate=<path to your certificate in PEM format>
http-api-private-key=<path to your RSA private key file PEM format>
```

The certificate must contain the certificate itself and the intermediate certificates of your CA. A description how to bundle the certificates into one file can be found here: https://support.globalsign.com/ca-certificates/root-certificates/root-intermediate-certificate-bundles

2. Use a web-server, like the Apache, as a front-end for the ssl communication and forward the requests to the agent using the apache proxy module. This is the recommended configuration in case of using the inbox-functionality. Set

http-api-type to *tcp* and start the agent using the standard port. This is an example apache configuration which must be added to the ssl / virtual host configuration an will forward requests to the agents standard port 45454:

```
<Location ~ ^/>
   RewriteEngine On
   # Replace <my-own-ip> with your static IP to disable
   # access to the agents login page for unknown users
   RewriteCond "%{REMOTE_ADDR}" "!<my-own-ip>"
   RewriteRule (/getLoginInformation|/views/login.html) - [L,R=403]
</Location>

# All requests will be forwarded to the standard port of the agent
# Replace the port number 45454 from your agent configuration
ProxyPass / http://127.0.0.1:45454/
ProxyPassReverse / http://127.0.0.1:45454/
```

CHAPTER

FIVE

PERFORMANCE CONSIDERATION

The TeamDrive Agent is typically used to synchronize data on a central server. These are some guidelines to optimize the performance of the TeamDrive Agent in this situation:

1. Do not mount the file system over the network, which contains the data that will be synchronized. Network shares are much slower than local file systems. Using a network share will limit the number of files and directories per Space.
2. Use more than one TeamDrive Space. Using more than one Space can increase the performance. For example, you could have one Space per project or team.

CHAPTER

SIX

UPDATING FROM THE TEAMDRIVE 3 AGENT TO THE TEAMDRIVE 4 AGENT

The application data directory (specified by the `teamdrived --teamdrive-home` (page 21) parameter) is compatible to TeamDrive 3 and will be updated in this process.

1. Perform a full system backup in order to be able to restore your system if you encounter unexpected errors. if you encounter unexpected errors. In case you decide to skip the full system backup, it is of particular importance that you backup the TeamDrive 3 Agent application data directory. The TeamDrive 3 Agent application data directory is located in the 'AppData\Roaming' folder of the Windows user used to install the Agent. You can navigate to the folder using "%ProgramData%" → "TeamDrive3" → "teamdrive.ini" or simply *C:\ProgramData\TeamDrive3\teamdrive.ini*. Open the file with a text editor, locate the entry "teamdrive-home-agent=", copy and navigate to this path and simply backup the "TeamDrive3" folder.
2. Shutdown the TeamDrive 3 Agent.
3. Install the new TeamDrive 4 Agent as described, but do not start the TeamDrive Agent.
4. Make sure that you have only one installation of the TeamDrive Agent. Manually delete older TeamDrive 3 installations in case they are not automatically deleted. Keep in mind that the application data directory will be used by TeamDrive 4 afterwards. Older TeamDrive 3 installations can be found in *C:\Program Files(x86)*. Manually delete old TeamDrive 3 installations by simply deleting their folders.
5. Start the TeamDrive 4 Agent. See *Initial Set-Up* (page 13).

> **Warning:** On Windows, the *uninstall.exe* will delete the application data directory. Be careful when removing old installations.

CHAPTER

SEVEN

TROUBLESHOOTING

TeamDrive stores log output to %APPDATA%\TeamDrive3\logs or ~/.teamdrive/logs. Please search for [Critical] messages that can be a hint to an error.

7.1 Administration interface is not available (Linux)

If you are not able to access the administration interface in you browser, please check that TeamDrive has opened a TCP port:

```
[root@agent ~] netstat -twaunp
```

Please edit the settings file and set TeamDrive to open the port 45454. This can be done by editing the file ~/.teamdrive/teamdrive.settings file. This is an example:

```
[root@agent ~] vim ~/.teamdrive/teamdrive.settings
```

Add the following entries:

```
[Settings]
http-api-type=tcp
http-api-port=45454
```

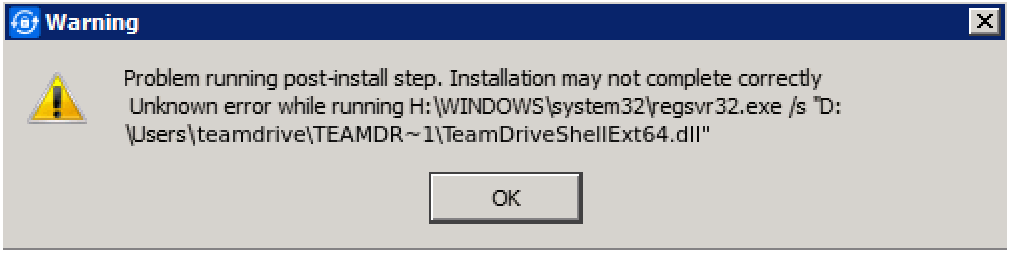
Restart TeamDrive with the following commands:

```
[root@agent ~] killall teamdrived.bin
# Wait 10 seconds
[root@agent ~] ./TeamDrive
```

TeamDrive stores log output to %APPDATA%\TeamDrive3\logs or ~/.teamdrive/logs. Please search for [Critical] messages that can be a hint to an error.

7.2 Installation error: Probelm running post-install step (Windows)

This issue is related to how a Windows Terminal Server works. It is redirecting the CSIDL_WINDOWS path to user's home directory such as C:\Users\username\Windows. This can be solved by switching the machine to install mode before running the installer, whoch can be done by running the Programs and Features from *Control Panel* and choosing to *install an application in terminal server*. It can also be done by running change user /install command from command prompt and change it back to execution mode by running change user /execute after the installer finishes.



Warning

Problem running post-install step. Installation may not complete correctly
Unknown error while running H:\WINDOWS\system32\regsvr32.exe /s "D:
\Users\teamdrive\TEAMDR~1\TeamDriveShellExt64.dll"

Fig. 7.1: Error message shown by the installation process on a Windows Terminal Server.

7.3 Using a system proxy in Windows

Please notice, that services will start under the LOCAL SYSTEM user and have no access to the proxy settings of a standard windows user. You will find different solutions how to set a system proxy for services:

http://serverfault.com/questions/34940/how-do-i-configure-proxy-settings-for-local-system

If you want to copy the proxy settings from a TeamDrive Desktop client instance using a system proxy to a TeamDrive Agent instance, set the proxy type to 3 (for manual proxy):

Type=3

in case that you dont want to configure the system proxy as described in the above article. Otherwise the TeamDrive Agent will override the setting with the not configured empty system proxy values.

CHAPTER

EIGHT

SYNOPSIS

teamdrived [options] (autostart | shutdown)

CHAPTER

NINE

OPTIONS

This is a list of known settings or command line arguments that are used by TeamDrive.

9.1 Local Installation Settings

--spaces-path

Default path for newly created Spaces by the user.

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `C:/Path/to/Spaces`.

--teamdrive-home

Set TeamDrive application data path. Typically `~/.teamdrive` on Linux or Mac OS and `%APPDATA%\TeamDrive3` on Windows.

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `teamdrive-home=C:/Path/to/Appdata`.

--data-path

Default location for temporary data and cache files. Defaults to *teamdrived --teamdrive-home* (page 21)

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `data-path=C:/Path/to/TeamDrive Data`.

--default-server

Default host server to create spaces on

--perform-backup=`true/false (default: true)`

Enable / Disable creation of Backups

--backup-path

Set the Backup path

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `backup-path=C:/Path/to/Backups`.

--language
Set the user interface language. This setting is deprecated.

Deprecated since version 4.1.3.

--autostart-delay
Delay background tasks to speedup system startup time

--working-set-size
Maximum cache size

--enable-folder-decoration=true/false (default: true)
Enable / Disable file system folder decoration

--enable-create-favorite-link-to-spaces=true/false (default: true)
Windows only: creates a shortcut in the Windows explorer favorites to the *teamdrived --spaces-path* (page 21) folder.

--device=<device>
if *device* is "softwarecontext", TeamDrive will use the QtQuick2dRenderer as the rendering backend (Windows only)

--enable-multi-processes=true/false (default: false)
Allow spawning of multiple TeamDrive instances.

--enable-shell-extension=true/false (default: true)
Enables or disables the Windows Explorer integration or the Finder Extension.

New in version 4.1.2.

autostart
Don't show the main window.

shutdown
Shutdown the currently running TeamDrive process.

webgui
Lauches TeamDrive with a WebGUI instead of the normal user interface.

Note: Desktop only.

9.2 Registration Server Communication

--check-for-updates=true/false (default: true)
The TeamDrive Client will check for software updates on the Registration Server. Set this value to false, if a software distribution tool will be used to deploy Client installations to your users.

--auto-accept-invitation=true/false (default: false)
When set to true, the TeamDrive Client will accept all Space invitations automatically and join these Spaces.

--auto-accept-invitation-mode (default: archived)
The mode of operation when joining Spaces automatically. Possible values are: current-version-only, all-versions, directory-information-only, no-sync-to-filesystem, offline-available, archived.

--auto-invite-users=list
A list of user names to be automatically invited into newly created Spaces with specified DefaultInvitationRights. The list has to be separated by semicolons and enclosed with double quotes in the settings file. Example: auto-invite-users="abc;def"

--enable-key-repository=true/false (default: true)
Enable / Disable the Key Repository.

9.3 Synchronization Settings

--default-file-permissions
Default unix file permissions. Default is 644.

--default-dir-permissions
Default unix directory permissions. Default is 755.

--scan-enabled=true/false (default: true)
On startup, the internal database will be compared with the file system using a file system scan to detect Space changes while TeamDrive was not running.

--default-publish-expiry-days (default: 0)
Default value for the number of days, a file is published. Requires a *HostServer* Version >= `3.5`. A value of `0` means *unlimited*.

9.4 Agent Settings

--http-api-port=<host>
where <*host*> equals one of the following:

1. a port number (e.g. `45454`)
2. an IPv4 address including the port number (e.g. `127.0.0.1:45454`)
3. an IPv6 address including the port number (e.g. `[::0]:45454`)
4. a pipe or socket name. Requires `http-api-type=local`.
5. a pipe or socket name. Requires `http-api-type=local`.

--http-api-type
Sets the type of the HTTP API.

1. `local`: Opens a Named pipe on Windows or a Socket file on Linux
2. `tcp`: Opens a TCP port.
3. `ssl`: Opens a SSL/TLS encrypted TCP socket. See *teamdrived --http-api-private-key* (page 23) and *teamdrived --http-api-certificate* (page 23)

--http-api-certificate
SSL-Server only: A path to a certificate file for the api (PEM format).

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `http-api-certificate=C:/Path/to/Certificate.pem`.

--http-api-private-key
SSL-Server only: A path to a RSA private file for the certificate (PEM format). See also *teamdrived --http-api-certificate* (page 23).

Note: On Windows, this path may not contain backslashes. Please use slashes instead like this: `http-api-private-key=C:/Path/to/Key.key`.

`--idle-shutdown-timeout`

The number of idle seconds, until the TeamDrive Agent shuts itself down. Typically used with automatically shutdown of Linux containers. Default: no timeout.

New in version 4.1.1.

`--http-api-external-register-url`

This url will be used to show a "Register Now" link in the Web UI. This setting is mainly useful for Web Portals.

New in version 4.1.1.

`--http-api-external-login-url`

This url will be used to redirect the Browser if a login is required. This setting is intended to be used for external authentication.

New in version 4.1.1.

`--http-api-compress-replies``=true/false (default: true)`

Use a zipped HTTP content-encoding for JSON replies.

New in version 4.1.1.

9.5 Platform specific Settings

It is possible to limit a setting to a specific domain. For example if you add -linux to a setting, this setting will only affect Linux clients.

Suffix	Description
-mac	Mac OS X only.
-win	Windows only.
-linux	Linux Clients only.
-agent	Used by the Headless Agent.

CHAPTER TEN

CONFIGURATION FILES

TeamDrive reads settings from these locations:

1. Program arguments as mentioned above.
2. The read-only `DISTRBIBUTOR` file in the application installation path, eg. `C:\Program files\TeamDrive\DISTRIBUTOR` or `/opt/teamdrive/DISTRIBUTOR`.
3. A system wide configuraion file. Windows: `%ProgramData%\TeamDrive3\teamdrive.ini`, Linux: `/etc/teamdrive.ini`, Mac OS: `/Users/Shared/teamdrive.ini`.
4. The `teamdrive.settings` file in the application data directory, eg. *teamdrived --teamdrive-home* (page 21). Typically `~/.teamdrive` on Linux or Mac OS and `%APPDATA%\TeamDrive3` on Windows.
5. The Registration Server may also provide some of these settings. Please consult the Registration Server documentation for details.

The majority of arguments can also be added to **[Settings]** Section of the configuraion files.

This is a valid `teamdrive.settings` file that sets two settings on Linux:

```
[Settings]
language=en-US
spaces-path=/home/teamdrive/Spaces
```

And this is the corresponding settings file for Windows:

```
[Settings]
language=en-US
spaces-path=C:/Users/TeamDrive/Documents/Spaces
```

Please note the usage of forward slashes instead of backslashes on Windows. Also notice that a setting in the `teamdrive.settings` file is not prefixed by two dashes (`--`).

CHAPTER

ELEVEN

TEAMDRIVE HTTP API

11.1 Overview

The *TeamDrive Agent* offers a JSON interface via HTTP, which allows you to operate the *TeamDrive Agent* without using the GUI. The JSON interface can be activated by specifying the `teamdrived --http-api-port` (page 23) option.

The command line application *TeamDriveApi.py* is located in the *TeamDrive Agent* installation directory (e.g. *C:\Program Files (x86)\TeamDrive 3 Agent*). Running this file allows you to use the TeamDrive API directly. See *Options* (page 21) for a list of Options and Settings.

If you would like to use the TeamDrive HTTP API without Python, HTTP *GET* requests can be made directly through *TeamDrive Agent*. The URL will need to be constructed as follows:

```
http://<host>/api/<command>?<parameters>
```

For example:

```
http://[::1]:45454/api/getSpace?id=1
```

The result is a JSON document.

11.2 Requirements

In order to run use the interface, you will need the these requirements:

- TeamDrive Desktop Client or *TeamDrive Agent*

The `Python module and Command line interface` requires

- *Python* in your *PATH* variable.
- Recommended: docopt for Python.
- `TeamDriveApi.TeamDriveApi.sse()` (page 72) requires *Python 3*.

11.3 Security Considerations

Communicating with the *TeamDrive Http Api* contains sensitive information, for example user names, passwords and file data. There are mechanisms to provide authentication and privacy.

11.3.1 Restricting the availability of the Api

By restricting the api to the local host, any attempt to access the api from another host will fail:

```
http-api-port=127.0.0.1:<port>
```

Although, this does not prevent the usage of the api from another user of this system.

11.3.2 Using a local socket or a named pipe

One can further reduce the availability by specifying the type of the api like this:

```
http-api-type=socket
http-api-port=<name>
```

<name> can be a user defined string. The Unix socket file will be placed in */tmp* on Mac OS X and on Linux.

11.3.3 Using HTTPS for communication with the TeamDrive Agent

HTTPS can be enabled by setting *type* to *ssl* like this:

```
http-api-type=ssl
http-api-port=443
```

It is recommended to use port 443 for communication with browsers, because using a non-standard port for HTTPS is considered insecure. Please also have a look at the `teamdrived --http-api-certificate` (page 23) and `teamdrived --http-api-private-key` (page 23) settings.

Note: The HTTPS server that is used by the TeamDrie Agent is based on the Qt framework, which supports, according to the documentation, the following protocols: `SSLv2`, `SSLv3`, `TLSv1.0`, `TLSv1.1` and `TLSv1.2`.

11.4 Authentication

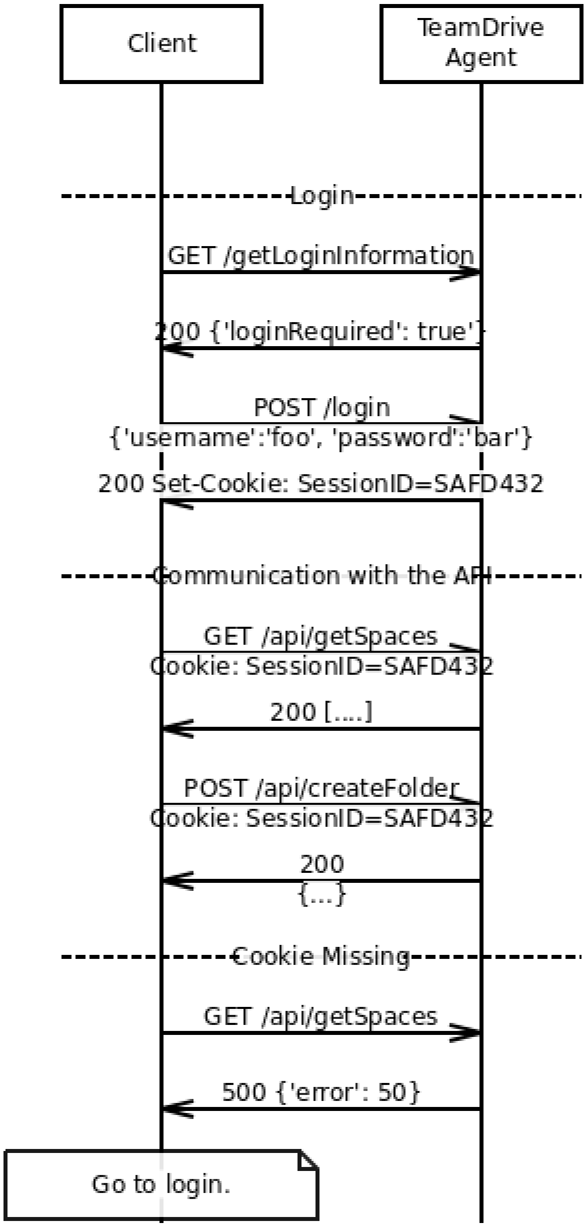
In order to prevent unauthorized users from accessing the *TeamDrive Http Api*, an authentication is **required** to access the api. There are two ways to authenticate clients:

1. Statefull Session Cookies
2. Stateless Authorization Header

11.4.1 Session Cookies

Session Cookies are mainly used to authenticate Web Browsers. Using a session requires a call to `TeamDriveApi.TeamDriveApi.login()` (page 60) to retrieve a cookie from the server. `TeamDriveApi.TeamDriveApi.getLoginInformation()` (page 49) can be used to determine if a login is required.

Login with the Api using Cookies should follow this protocol:



In the case of a missing Cookie, a *TeamDriveApi.TeamDriveException.Error_Missing_Session* (page 78) is returned

11.4.2 Authorization Header

This stateless alternative to Cookies uses HTTP Basic Authentication. **Every** request to the Api requires an *Authorization Header* following **RFC 2617**.

Currently, The Agent does *not* answer requests without Authentication Header appropriately. It is expected, that clients send the Authorization Header unconditionally.

11.5 Interface Overview

TeamDrive provides two API endpoints:

- The `/api` endpoint provides meta data about all elements of TeamDrive
- The `/files` endpoint provides an interface for reading and writing the file content.

11.5.1 Communicating with the /api endpoint

This is the reference documentation of the TeamDrive Client API. As a convenience, there is a Python module that provides accessor functions for each API call, which is also usable as a command line interface. Please have a look at all possible usages.

Example using curl:

```
$ curl '127.0.0.1:45454/api/getLoginInformation'
{ "isLoginRequired" : true }
```

Example using the Python module:

```
>>> from TeamDriveApi import TeamDriveApi
>>> api = TeamDriveApi("127.0.0.1:45454")
>>> map(api.getSpace, api.getSpaceIds())
[{.....}]
```

Example using the command line:

```
./TeamDriveApi.py '127.0.0.1:45454' getLoginInformation
{
    "username": "My Username",
    "apiUrl": "/",
    "isLoginRequired": false
}
```

Example Session (Initial Login):

```
# Start TeamDrive:
exec /opt/teamdrive/teamdrived

# Wait for it:
sleep 5

# Log-in
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass login

# Create a new Space
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass createSpace

# Call Functions
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass createFolder

# Quit TeamDrive without logout
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass quit False
```

Example Session:

```
exec /opt/teamdrive/teamdrived
sleep 5

# No need to login now
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass createFolder

# Quit TeamDrive without logout
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass quit False
```

Example Session (After previous Logout:

```
# If the previous session ended with a logout:
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass quit True

# The next call should start with a login:
exec /opt/teamdrive/teamdrived
sleep 5
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass login
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass createFolder
```

See also:

`TeamDriveApi.TeamDriveException` (page 76) for Errors returned by the API.

See also:

Chapter *Authentication* (page 27) regarding Authentication.

11.5.2 Communicating with the /files endpoint

See `TeamDriveApi.TeamDriveApi.putFile()` (page 63) for Details.

11.5.3 Passing Parameters to the API

The format of parameters depend on the HTTP method:

- Parameters for GET-requests should be passed by a query string in the url. For example `curl --user myTeamDriveUser http://127.0.0.1:45454/api/getSpace?id=310`
- Parameters to POST-requests can either be a JSON object or data encoded using `application/x-www-form-urlencoded`.

11.6 Interface Description

11.6.1 TeamDriveApi.TeamDriveApi

class `TeamDriveApi.`**`TeamDriveApi`**(*server=’[::1]:45454’*, *username=’‘*, *password=’‘*)

Return type *TeamDriveApi* (page 30)

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.

- **ConnectionRefusedError** – a Connection Error.

about ()

Returns information about TeamDrive.

Returns

```
{
  "version": "4.0.12 (Build: 1294)",
}
```

Changed in version 4.1.1: no longer returns license infos. call *getLicense* (page 49) instead.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

addAddressbook (*name*)

Adds a new Address book entry. TeamDrive will lookup that name on the RegServer. If that fails, an exception is thrown. You cannot add email addresses.

Using this function requires the HTTP POST method.

Parameters **name** (*str*) – the new Addressbook's name.

Returns the id of the new entry.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – user is unknown.
- **ValueError** – no JSON returned. Wrong URL?

addComment (*fileId*, *comment*, *notifyMembers*)

Attaches the comment to the file.

Using this function requires the HTTP POST method.

See also:

getComments (page 45), *getSpaceComments* (page 53), *addSpaceComment* (page 32) and *removeComment* (page 65)

New in version 4.2.1.

Parameters **notifyMembers** (*bool*) – Send notification to all members (default is false)

New in version 4.1.1.

Parameters

- **fileId** (*int*) – The corresponding id of the file

- **comment** (*str*) – The content of the new comment

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Adding this comment failed.
- **ValueError** – no JSON returned. Wrong URL?

addServer (*url*, *name*, *username*, *password*, *path*, *type*)

Register a WebDAV or TDPS server for use by the TeamDrive client.

Parameters

- **url** (*str*) – Url of the server
- **name** (*str*) – Name of the server in TeamDrive
- **username** (*str*) – WebDAV only: username for the server
- **password** (*str*) – WebDAV only: password for the server
- **path** (*str*) – WebDAV only: path to the server
- **type** (*str*) – 'TDPS' or 'WebDAV'

Return type dict

addSpaceComment (*spaceId*, *comment*, *notifyMembers*)

Adds this comment to the Space.

Using this function requires the HTTP POST method.

See also:

getComments (page 45), *getSpaceComments* (page 53), *addComment* (page 31) and *removeComment* (page 65)

New in version 4.2.1.

Parameters **notifyMembers** (*bool*) – Send notification to all members (default is false)

New in version 4.1.1.

Parameters

- **spaceId** (*int*) – The Space id.
- **comment** (*str*) – The content of the new comment

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Adding this comment failed
- **ValueError** – no JSON returned. Wrong URL?

cancelProxyAuthentication()

Cancel Proxy-Authentication process.

Return type dict

check2FA(*otp='', authType='email-otp'*)

Check whether the user has completed two-factor-authentication yet, optionally sending a one-time-pin (OTP) entered by the user along with the request. (The PIN is optional because you might want to use this function to check whether the user has sent the PIN to the server via some other channel, eg. an authentication link).

The `authType` parameter indicates which 2FA method is being used (see :any:`start2FA``).

Parameters

- **otp** (*string*) – (Optional) th.
- **authType** (*string*) – 2FA method being used (defaults to "email-otp")

Return type dict

clearProfilePicture()

Clears the profile picture of the current user.

See also:

getProfilePicture (page 50) to retrieve profile pictures and *setProfilePicture* (page 70)

New in version 4.1.1.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

commitCurrentSnapshot(*spaceId*)

Commit the snapshot that is currently being previewed.

The process for restoring a Space to a snapshot is as follows:

1. 'preview' a snapshot by calling the *joinSpace* (page 58) function with the 'snapshotId' parameter
   - The Space is reverted *locally* to the state stored in the snapshot, in read-only mode
   - The user can then decide whether they want to commit this change.
2. 'commit' the change by calling *commitCurrentSnapshot* (page 33)
   - The Space is reverted on the server (this affects not only files but also the members in the Space, member permissions, etc.)

While in preview-mode, you can also call *joinSpace* (page 58) again with a different snapshot ID to preview another snapshot, or without a snapshot ID to cancel the rollback and restore the Space to its usual state.

Using this function requires the HTTP POST method.

New in version 4.3.2.

Parameters **spaceId** (*int*) – The Space id.

Return type bool

`confirmSuperPinExport`()

This function marks the super-PIN/account recovery data as "exported" - ie. the user has stored it in some safe place. With the recovery data marked as "exported", TeamDrive will stop reminding the user to take care of this task.

Return type dict

`createBackup`()

Create a backup in the local file system. To create a backup and download it, send a GET request to '/backup'. To import a TeamDrive backup file ('.tdbk'), send it to '/backup' via a PUT request.

Returns A dict where the key "path" points to the file in the local filesystem.

New in version 4.3.1.

Return type dict

`createFolder`(*spaceId*, *filePath*)

Creates a new folder in that Space

Example:

```
$ ./TeamDriveApi.py '127.0.0.1:45454' createFolder 3 '/my new folder'␣
↪False
```

```
{
  "result": true,
  "file": {
        "isDir": true,
        "name": "my new folder",
        "currentVersionId": 0,
        "spaceId": 3,
        "isInFileSystem": false,
        "path": "/",
        "id": 756,
        "permissions": "rwxr-xr-x"
  }
}
```

See also:

`getFullFolderContent` (page 48) or *`getFolderContent`* (page 46) for getting the content of this folder

New in version 4.0.

Returns A dict containing a *File* (page 79) object

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`filePath`** (*`str`*) – The full folder path.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

- **ValueError** – no JSON returned. Wrong URL?

createInbox (*folderId*, *expiryTime=None*, *password=None*, *description=None*, *maxFileCount=None*, *maxFileCountInterval=None*, *maxFileSize=None*, *maxSize=None*)

Convert the given folder into an inbox, allowing anybody with the link to the inbox to upload files into the folder (the inbox service needs to already be set up in order to enable this functionality).

Parameters

- **folderId** (*int*) – The id of the folder to convert to an inbox
- **expiryTime** (*str*) – Optional date on which the inbox-access to this folder will automatically be disabled again, for example "2020-09-30T10:52:15.762Z".
- **password** (*str*) – If set, the password people need to enter before uploading to the inbox.
- **description** (*str*) – Optional description for the inbox
- **maxFileCount** (*int*) – Optional maximum number of files that can be uploaded to the inbox in the interval specified by `maxFileCountInterval` (Only has an effect if `maxFileCountInterval` is not set to "Unlimited")
- **maxFileCountInterval** (*str*) – One of "Unlimited", "Infinite", "Day", "Week", "Month", "Quarter", "HalfYear", "Year". For example, if maxFileCount is set to 10, and this parameter is set to "Week", at most 10 files may be uploaded to the inbox per week. If this parameter were set to "infinite" instead, then the inbox would only accept at most 10 files in total. The default is "Unlimited".
- **maxFileSize** (*int*) – Optional limit on the size of individual uploaded files, in bytes.
- **maxSize** (*int*) – Optional limit on the total file size of all files uploaded to the inbox, in bytes.

Return type dict

createSnapshot (*spaceId*)

Create a snapshot that can later be used to restore the Space (see `commitCurrentSnapshot` (page 33))

Using this function requires the HTTP POST method.

New in version 4.3.2.

Parameters **spaceId** (*int*) – The Space id.

Return type bool

createSpace (*spaceName*, *disableFileSystem=None*, *spacePath=None*, *importExisting=None*, *serverId=None*, *inviteOwnDevices=None*, *versionCount=None*, *webAccess=None*, *dataRetentionPeriod=None*, *dataRetentionAutoTrash=None*, *dataRetentionDeleteGuard=None*, *defaultJoinMode=None*, *mountInVirtualFS=None*)

Creates a new Space with the given name.

To convert an existing folder into a TeamDrive Space, you need to set the *importExisting* parameter to True.

In case the url is called directly, the default for *disableFileSystem* is False and the *spacePath* is mandatory.

"Data Retention" Spaces

Spefifying the `dataRetentionPeriod` parameter creates a special "Data Retention" Space. Any file in the Space will not be deletable by anyone except the Space Administrator until the file is older than the specified retention period. This can be used to satisfy legal requirements that documents be retained for a minimum length of time.

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails.

returns A dict containig the Id and a *Space* (page 78).

Parameters

- **`spaceName`** (*`str`*) – The Name of the new Space
- **`disableFileSystem`** (*`bool`*) – This will disable the synchronization into the file system. TeamDrive will only synchronize meta-data if this is *True*.
- **`spacePath`** (*`str`*) – File Path, in which the new Space will be created.
- **`importExisting`** (*`bool`*) – Tells the TeamDrive Agent to import an existing directory.
- **`serverId`** (*`int | None`*) – Hostserver to use
- **`inviteOwnDevices`** (*`bool | None`*) – Whether to automatically invite your other teamdrive installations
- **`versionCount`** (*`int | None`*) – Maximum number of versions of each file that will be kept
- **`webAccess`** (*`str | None`*) – Whether this Space may be accessed from the web (one of “true”, “false”, or “default”)

  New in version 4.3.2.
- **`dataRetentionPeriod`** (*`int | None`*) – Minimum length of time (in months) before files can be deleted

  New in version 4.6.3.
- **`dataRetentionDeleteGuard`** (*`bool | None`*) –

  By default, files can still be deleted within the “data retention” period after an additional password check.
  is set to `true`, such deletions will be prevented as well.

  New in version 4.6.3.
- **`dataRetentionAutoTrash`** – Whether to automatically move files to the trash at the end of the data retention period

New in version 4.6.3.

Parameters **`defaultjoinMode`** – How members should join this Space by default (ie. the option that people who are invited to the Space will see auto-selected). One of: * default -> The default settings of the invited user will be shown as auto-selected * offline-available -> By default, the Space will be joined in offline-available mode (ie. all files downloaded locally) * non-offline-available * virtual -> Space will be mounted as a folder in the virtual file system (where possible)

New in version 4.6.7.

Parameters **`mountInVirtualFS`** – Whether to mount the Space as a folder in the virtual file system.

New in version 4.6.7.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Space creation failed.
- **ValueError** – no JSON returned. Wrong URL?

deleteFileFromTrash (*fileId*)

Deletes a file from Trash by physically deleting the file from the server. This call fails, if you do not have the rights to delete files, it is a directory or the file is not in the trash.

Using this function requires the HTTP POST method.

Example:

```
>>> api = TeamDriveApi()
>>> for inTrash in api.getFullFolderContent(1, "/", True):
>>>     api.deleteFileFromTrash(inTrash["id"])
```

Throws *TeamDriveException* (page 76), if the call fails.

See also:

removeLocallyFile (page 65) and *setTrashed* (page 72).

New in version 4.0.

Parameters **fileId** (*int*) – The Id of the Space

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

deleteInbox (*id*)

Delete the given inbox (ie. remove upload-access to the folder via the inbox URL, neither the folder itself nor any files in it will be affected).

Parameters **id** (*int*) – The id of the Inbox

Return type dict

deleteServer (*id*)

Delete a WebDAV or TDPS server.

Parameters **id** (*int*) – Id of the server

Return type dict

deleteSnapshot (*spaceId*, *snapshotId*)

Delete snapshots for this Space based on the given `snapshotId`

> **Warning:** This function deletes *all* snapshots up to and including the one specified by `snapshotId`

Using this function requires the HTTP POST method.

New in version 4.5.0.

Parameters

- **`spaceId`** (*`int`*) – The Space id.
- **`snapshotId`** (*`int`*) – The snapshot id.

Return type bool

`deleteSpace` (*id*, *delInFs*, *delOnServer*, *delInDB*)

Deletes a Space.

Using this function requires the HTTP POST method.

Throws *`TeamDriveException`* (page 76), if the call fails.

Parameters

- **`id`** (*`int`*) – The Id of the Space
- **`delInFs`** (*`bool`*) – Delete the Space in the File System
- **`delOnServer`** (*`bool`*) – Deletes the Space on the server. (requires Administrator rights)
- **`delInDB`** (*`bool`*) – Deletes the Space in the local Database

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Deletion failed
- **`ValueError`** – no JSON returned. Wrong URL?

`deleteVersion` (*versionId*)

Deletes a file version finally from the server. This call fails, may may not have the rights to delete files,

Using this function requires the HTTP POST method.

Throws *`TeamDriveException`* (page 76), if the call fails.

See also:

`getVersion` (page 57)

New in version 4.1.4.

Parameters **`versionId`** (*`int`*) – The Id of the version

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

dismissNewFiles (*spaceId=0*, *versionIds=[]*)

'dismiss' new files so that they will no longer appear when using the `new` filter with the function *findFiles* (page 39). All new files can be dismissed by setting the `spaceId` parameter, or specific file versions can be dismissed via the `versionIds` parameter - exactly one of the parameters must be set.

New in version 4.7.6.

Parameters

- **spaceId** (*int*) –
- **versionIds** (*list[int]*) –

Return type dict

editServer (*id*, *url*, *name*, *username*, *password*, *path*)

Edit a WebDAV or TDPS server.

Parameters

- **id** (*int*) – Id of the server
- **url** (*str*) – Url of the server
- **name** (*str*) – Name of the server in TeamDrive
- **username** (*str*) – WebDAV only: username for the server
- **password** (*str*) – WebDAV only: password for the server
- **path** (*str*) – WebDAV only: path to the server

Return type dict

emptyTrash (*spaceId*)

Permanently delete all trashed files for the given Space.

Note: Depending on the number of files / file versions, the contents of the trash may not immediately disappear after this function returns

Using this function requires the HTTP POST method.

New in version 4.5.3.

Parameters **spaceId** (*int*) – The Space id.

Return type dict

exportRecoveryData ()

Export recovery data (the "super PIN" and a recovery-URL).

Return type dict

findFiles (*spaceIds=None*, *path='/'*, *search=None*, *recursive=False*, *filters=None*, *sortBy='name'*, *start=0*, *limit=-1*)

This function can be used to create a paged list of files in all Spaces, a single Space, or a single folder.

The result is returned like this:

```
{
        totalFiles: 123,
        files: [...]
}
```

`totalFiles` is the number of files that were matched, `files` is a subset of the matched files based on the `start` and `limit` parameters. See *`getFile`* (page 45) for the format of the individual file objects.

The parameter `filter` is a comma-seperated list of strings indicating special filters. Valid values are:

- `conflicted`: Show only files that have name- or version conflicts
- `directory`: Show only folders
- `trashed`: Only files that are in the trash
- `invalidPath`: Only files that have an invalid path or name for the OS that the agent is running on
- `offlineAvailable`: Show only files that have been set to "offline-available"

The above filters can also be prefixed with a "!" to invert them, eg. `!trashed` to show only files that are *not* in the trash

Additionally, *one* of the following can be included in the filter list (these values can't be prefixed with "!"):

- `unconfirmed`: Show only files where the latest version has not been processed by TeamDrive yet
- `commented`: Show only files that have comments
- `published`: Show only published files
- **`new`: Show only files that have recently been added to the Space, or, in Spaces that have read-confirmations enab**
  New in version 4.7.6.

The returned list is sorted according to the `sortBy` parameter, which contains a list of file properties that are considered in order (eg. specify `[isDir,name]` to sort by name, but list directories first). Valid values in the list are:

- `isDir`: List directories first
- `name`: Sort by file name
- `added`: sort by the time this file was added to the Space
- `created`: Sort by date of file creation
- `size`: Sort by size (directories considered as having a size of 0)
- `modified`: Sort by last-modified date (directories are sorted as though the date was 0)

All of the sort-types are in ascending order by default, and can be inverted by prefixing them with "!", eg. `!size` to list in order of descending size.

New in version 4.5.1.

Parameters

- **`spaceIds`** (*`list[int]`*) – The IDs of the Spaces to search (default if not specified: search all Spaces)
- **`path`** (*`str`*) – The path in the Space(s) under which to search for files. (Default is '/')
- **`search`** (*`str`*) – If set, files will be filtered by matching the file name against this parameter
- **`recursive`** (*`bool`*) – Whether to recursively search subfolders of the given path (`False` by default)

- **filters** (*list[str]*) – A comma-separated list of special filters. See the descriprion of this function for details
- **sortBy** (*list[str]*) – A list of file properties to sort by, see the description of this function for details (default is `name`)
- **start** (*int*) – The first row of the matched set of files to return (0-indexed, the default if not specified is `0`)
- **limit** (*int*) – Maximum number of rows to return, or `-1` to return all rows (default is `-1`)

Returns A dict with the number of files matched, and the subset of those files specified by `start` and `limit`

Return type dict

findUser(*query*)

Returns information on a TeamDrive user, queried either by name or email address (the TeamDrive user must already be in the local address book).

Parameters **query** (*str*) – Name or Email

Return type dict

getActivities(*spaceIds=None*, *fromDate=None*, *toDate=None*, *searchTerm=None*, *typeFilter=None*, *userFilter=None*)

Returns a list of all activities for the given space list within the date period or for the last 30 days. If `hasDetails` is `true`, a call to *getActivityDetails* (page 43) with the given activityId will return a list of details.

Possible activity types are:

- `userInvited`
- `userJoined`
- `userLeft`
- `userKicked`
- `commentCreated`
- `commentDeleted`
- `spaceCommentCreated`
- `spaceCommentDeleted`
- `versionAdded`
- `versionDeleted`
- `versionPublished`
- `versionUnpublished`
- `versionExpired`

Changed in version 4.2.0.

Added new activity types:

- `userRightsChanged`
- `userProfileUserNameChanged`
- `userProfileEmailChanged`
- `userProfilePhoneChanged`
- `userProfileMobileChanged`
- `userProfileNotesChanged`
- `userProfilePictureChanged`
- `userProfileInitialsChanged`
- `userLicenseChanged`
- `userRegEmailChanged`

- `spaceMaxVersionsOnServerChanged`
- `spaceMetaDataChanged`
- `spaceMetaDataDeleted`
- `spaceDeletedOnServer`
- `serverAccessSent`
- `serverAccessFailedToSend`
- `serverAccessReceived`
- `serverAccessDeleted`
- `invitationSent`
- `invitationFailed`
- `invitationDeclined`
- `invitationReceived`
- `fileAdded`
- `fileRenamed`
- `fileMoved`
- `fileTrashed`
- `fileUntrashed`
- `fileDeleted`
- `fileChanged`
- `fileCurrentVersionChanged`
- `folderAdded`
- `folderRenamed`
- `folderMoved`
- `folderTrashed`
- `folderUntrashed`
- `folderDeleted`
- `folderChanged`
- `fileNameConflict`
- `fileNameConflictResolved`
- `filePathInvalid`
- `filePathInvalidResolved`
- `versionPublishMetaChanged`
- `versionConflict`
- `versionConflictResolved`
- `versionExported`
- `emailNotificationSent`
- `emailNotificationFailedToSend`
- `workingDirFull`
- `runningOutOfDiskSpace`
- `noDiskFreeSpace`
- `automaticTrashCleanupChanged`
- `temporaryError`

See also:

getActivityDetails (page 43)

New in version 4.1.1.

Returns A list of activities.

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.

- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

Changed in version 4.2.0.

Parameters

- **spaceIds** (*str*) – A list of space IDs. If empty, all Spaces will be used.
- **fromDate** (*str*) – ISO 8601 extended format: either YYYY-MM-DD for dates or YYYY-MM-DDTHH:mm:ss, YYYY-MM-DDTHH:mm:ssTZD (e.g., 1997-07-16T19:20:30+01:00) for combined dates and times. If empty the last 30 days will be returned.
- **toDate** (*str*) – ISO 8601 extended format: either YYYY-MM-DD for dates or YYYY-MM-DDTHH:mm:ss, YYYY-MM-DDTHH:mm:ssTZD (e.g., 1997-07-16T19:20:30+01:00) for combined dates and times.

New in version 4.3.2.

Parameters

- **searchTerm** (*str*) – Search for activities that match the given string
- **typeFilter** (*str*) – Comma-separated combination of: `conflictAndError`, `invitationAndMember`, `fileAndVersion`, `comment`, `publish`, `all`
- **userFilter** (*str*) – one of: `myUser`, `otherUsers`, `all`

getActivityDetails(*activityId*)

Returns a list of all activity details for an activity.

See also:

getActivities (page 41)

New in version 4.1.1.

Parameters **activityId** (*str*) – The activityId as returned by *getActivities* (page 41)

Returns A list of activityDetails.

Return type list[dict]

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getActivityMetas(*activityId*)

Returns a list of all activity meta information for an activity.

See also:

getActivities (page 41)

New in version 4.2.1.

Parameters **activityId** (*str*) – The activityId as returned by *getActivities* (page 41)

Returns A list of activityMeta.

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getAddressbook`(*id*)

Returns information about a given Addressbook Id.

See also:

`getFullAddressbook` (page 47) for an example

Parameters **`id`** (*`int`*) – the Address Id.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getAddressbookByName`(*addressName*)

This is convenience wrapper of the Python module that is not part of the official API.

Returns an Addressbook entry by a given user name. Throws a TeamDriveException, if there is no Space with this name.

Parameters **`addressName`** (*`str`*) – The Name.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveCallFailed`** – There is no Space with this name
- **`ValueError`** – no JSON returned. Wrong URL?

`getAddressbookIds`()

Returns all known Addressbook Ids.

Changed in version 5.0: Added the `includeGroups` and `onlyDirectInvites` parameters.

Parameters

- **`includeGroups`** (*`bool`*) – Whether groups should be returned in the list. (Default is `False`)
- **`onlyDirectInvites`** (*`bool`*) – If `True`, members that were invited through a group and which have not yet joined the Space are omitted from the list. (Default is `False`)

Return type list[int]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getComments` (*fileId*)

Returns a list of all comments that are attached to a specific file in a space.

Example

```
>>> api = TeamDriveApi()
>>> api.getComments(4183)
```

Returns

```
[
  {
        "creator": "test",
        "creatorAddressId": 1,
        "text": "file baz",
        "created": "2015-09-01T15:15:31Z",
        "commentId": 16,
        "initials": "te"
  },
  {
        "creator": "test",
        "creatorAddressId": 1,
        "deleted": true,
        "created": "2015-09-01T15:15:28Z",
        "commentId": 15,
        "initials": "te"
  }
]
```

See also:

`getSpaceComments` (page 53), *`addComment`* (page 31), *`addSpaceComment`* (page 32) and *`removeComment`* (page 65)

New in version 4.1.1.

Parameters **`fileId`** (*`int`*) – The corresponding id of the file

Returns A list of comments.

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getFile` (*id*)

Returns information about the given File Id.

returns A *File* (page 79) object

Parameters **`id`** (*int*) – The File id

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getFiles` (*spaceId*, *filePath*, *trashed*)

Returns a list of files matching Space Id, file path and trashed flag.

returns A list of *File* (page 79) objects

Parameters

- **`spaceId`** (*int*) – The Id of the Space
- **`filePath`** (*str*) – the path of the file in the Space.
- **`trashed`** (*bool*) – Is the file in the trash?

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getFolderContent` (*spaceId*, *filePath*, *trashed*)

Returns a list of file Ids in a folder of a Space.

Example:

```
python ./TeamDriveApi.py '127.0.0.1:45454' getFolderContent 3 / False
[756,767,737,769,629,763,628,630,830,765]
```

Parameters

- **`spaceId`** (*int*) – The Id of the Space
- **`filePath`** (*str*) – The parent folder path.
- **`trashed`** (*bool*) – In Trash?

Return type list[int]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

- **`ValueError`** – no JSON returned. Wrong URL?

`getFolderInfo`(*spaceId*, *filePath*, *trashed*)

Returns information about the size of a folder:

```
{
  "files": 10,
  "size": 3288549
}
```

New in version 4.5.0.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`filePath`** (*`str`*) – The parent folder path.
- **`trashed`** (*`bool`*) – In Trash?

Return type list[int]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getFullAddressbook`()

Returns all Addressbook entries at once.

Example:

```
$ python ./TeamDriveApi.py '127.0.0.1:45454' getFullAddressbook
```

```
[
  {
        "id": 1,
        "email": "info@teamdrive.net",
        "name": "My Name",
        "icon": "self"
  }
]
```

Note: As this function was added with TeamDrive 4, the fall back for TeamDrive 3 is to iterate over all Addressbook Ids, therefore this function is much slower.

New in version 4.0.

Changed in version 5.0: Added the `includeGroups` and `onlyDirectInvites` parameters.

Parameters

- **`includeGroups`** (*`bool`*) – Whether groups should be returned in the list. (Default is `False`)
- **`onlyDirectInvites`** (*`bool`*) – If `True`, members that were invited through a group and which have not yet joined the Space are omitted from the list. (Default is `False`)

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getFullFolderContent` (*spaceId*, *filePath*, *trashed*)

Returns a list of file information in a folder of a Space.

Example:

```
$ python ./TeamDriveApi.py '127.0.0.1:45454' getFullFolderContent 3 /␣
↪False
```

```
[
  {
        "isDir": true,
        "name": "My folder",
        "currentVersionId": 0,
        "spaceId": 3,
        "isInFileSystem": true,
        "hasNameConflict": false,
        "hasVersionConflict": false,
        "path": "/",
        "id": 756,
        "permissions": "rwxr-xr-x"
  },
  {
        "isDir": false,
        "name": "My File",
        "currentVersionId": 0,
        "spaceId": 3,
        "isInFileSystem": true,
        "hasNameConflict": false,
        "hasVersionConflict": false,
        "path": "/",
        "id": 767,
        "permissions": "rwxr-xr-x"
  }
]
```

Note: As this function was added with TeamDrive 4, the fall back for TeamDrive 3 is to iterate over getFolderContent, therefore this function is much slower.

See also:

`getFile` (page 45) and *`getFolderContent`* (page 46)

New in version 4.0.

Returns A list of *File* (page 79) objects

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space

- **filePath** (*str*) – The parent folder path.
- **trashed** (*bool*) – In Trash?

Return type list[dict]

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getInbox(*id*)

Fetch information on a given Inbox.

Parameters **id** (*int*) – The id of the Inbox

Return type dict

getLicense()

Returns information about the current License.

Returns

```
{
  "distributor": "DOCU",
  "licenseKey": "DOCU-8782-5485-7266",
  "licenses": [
        "WebDAV",
        "Professional"
  ]
}
```

New in version 4.1.1.

See also:

setLicenseKey (page 68) to set the license.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getLoginInformation()

Returns information about the current user.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.

- **`ValueError`** – no JSON returned. Wrong URL?

`getMember`(*spaceId*, *addressId*)

Returns information about a member of a Space.

Example:

```
$ python ./TeamDriveApi.py '127.0.0.1:45454' getMember 3 1
```

Returns

```
{
        "status": "active",
        "spaceId": 3,
        "addressId": 1
}
```

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`addressId`** (*`int`*) – Already known Addressbook Id.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getProfilePicture`(*id*)

Returns the profile picture of the address.

See also:

`setProfilePicture` (page 70) to set the profile picture of the current user.

New in version 4.1.1.

Parameters **`id`** (*`int`*) – The addressId.

Returns The picture.

Return type bytes | dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

`getProxySettings`()

Get current proxy settings.

Return type dict

getReadConfirmations(*spaceId*, *addressId=0*, *fileId=0*, *filter='confirmed'*, *onlyLatest=True*, *fileSearch='`*, *limit=-1*, *start=0*)

Fetch "Read Confirmations" (or missing Read Confirmations) for a Space, file, folder or Space member.

The result is returned like this:

```
{
        totalConfirmations: 123,
        confirmations: [
                {
                        addressId : 1,
                        confirmed : true,  // would be False if this file␣
↪version hadn't been confirmed by this user
                        fileId : 142950,
                        name : "new.txt",
                        path : "/",
                        time : 1498746732698.0,
                        totalConfirmations : 1,   // # of confirmations for␣
↪this file, included if onlyLatest is True
                        versionId : 117039,
                        versionNo : 4
                },
                ...
        ]
}
```

New in version 4.5.1.

Parameters

- **spaceId** (*int*) –
- **addressId** (*int*) – Only get Read Confirmations for this user (optional)
- **fileId** (*int*) – Only get Read Confirmations for this file (optional)
- **filter** (*string*) – One of `confirmed`, `unconfirmed` or `all` to fetch Read Confirmations, missing cofirmations, or everything (defaults to `confirmed` if not set)
- **onlyLatest** (*bool*) – If `False` fetch every confirmation for every version of this file. Default is `True`: fetch only the latest confirmation per user and file
- **fileSearch** (*string*) – Filter by file name based on this string
- **limit** (*int*) – Fetch at most this many confirmations (default is `-1`, ie. no limit)
- **start** (*int*) – Index from which to start returning results (default is 0, use together with `limit` to implement paging)

Returns A dict with the number of confirmations matched, and the subset of those items specified by `start` and `limit`

Return type dict

getServer(*id*)

Returns information on a TeamDrive host server, a TDPS server, or a WebDAV server.

Parameters **id** (*int*) – The Id of the Server

Return type dict

getServerIds()

Returns a list of Host-, TDPS, and WebDav-Server ids.

Return type list[int]

getServers()

Returns a list of Host-, TDPS, and WebDav-Servers.

Return type list[dict]

getSettings()

Returns a *list* of settings, with each setting a *dict*. An incomplete list of settings can be found in the *Options* (page 21) Section.

Example :

```
{
        "key": "identifier",
        "label": "readable label",
        "help": "help text",
        "type": "string|path|integer|bool|enum|bitmask",
        "value": "current value",
        "options": ["enum option 1", "enum option 2"],
        "flags": [{bit: <bit flag>, label: "readable label"}, ...]
}
```

setSetting (page 71) can be used to set a specific setting.

Return type list

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getSnapshots(*spaceId*)

Get snapshots that can later be used to restore the Space (see *commitCurrentSnapshot* (page 33))

New in version 4.3.2.

Return type list[dict]

getSpace(*id*)

Will return information about that Space.

See also:

getSpaceStatistics (page 56) for statistics.

Example using Python:

```
>>> api = TeamDriveApi()
>>> map(api.getSpace, api.getSpaceIds())
{.....}
```

Example using curl:

```
curl --user myTeamDriveUser http://127.0.0.1:45454/api/getSpace?id=310

        :returns: A single :ref:`space-json-data-type`.
```

Parameters **id** (*int*) – The Id of the Space

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpaceByName`(*spaceName*)

This is convenience wrapper of the Python module that is not part of the official API.

Returns a Space by a given Space name. Throws a *`TeamDriveException`* (page 76), if there is no Space with this name.

Parameters **`spaceName`** (*`str | unicode`*) – The Name.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveCallFailed`** – There is no Space with this name
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpaceComments`(*spaceId*)

Returns a list of all comments that are attached to the Space itself.

Example

```
>>> api = TeamDriveApi()
>>> api.getSpaceComments(13)
```

Returns

```
[
  {
        "creator": "test",
        "creatorAddressId": 1,
        "text": "file baz",
        "created": "2015-09-01T15:15:31Z",
        "commentId": 16,
        "initials": "te"
  },
  {
        "creator": "test",
        "creatorAddressId": 1,
        "deleted": true,
        "created": "2015-09-01T15:15:28Z",
        "commentId": 15,
        "initials": "te"
  }
]
```

See also:

getComments (page 45), *addComment* (page 31), *addSpaceComment* (page 32) and *removeComment* (page 65)

New in version 4.1.1.

Parameters **spaceId** (*int*) – The Space id.

Returns A list of comments.

Return type list[dict]

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getSpaceIds()

Returns a list of all known Spaces Ids.

Example:

```
>>> api = TeamDriveApi()
>>> api.getSpaceIds()
[1,4,5,6]
```

Return type list[int]

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

getSpaceMemberIds(*spaceId*)

Returns all Addressbook Ids in a given Space.

Example:

```
$ python ./TeamDriveApi.py '127.0.0.1:45454' getSpaceMemberIds 3
[1,2]
```

New in version 4.0.

Deprecated since version 4.1.1: Use *getSpaceMembers* (page 55) instead.

Changed in version 5.0: Added the `includeGroups` parameter.

Parameters **includeGroups** (*bool*) – Whether groups should be returned in the list. If `True`, members that were invited through the group and have not yet joined the Space are omitted from the list. (Default is `False`)

Return type list[int]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpaceMembers`(*spaceId*)

Returns all Member infos in a given Space.

New in version 4.1.

Changed in version 5.0: Added the `includeGroups` parameter.

Parameters **`includeGroups`** (`bool`) – Whether groups should be returned in the list. If *true*, members that were invited through the group and have not yet joined the Space are omitted from the list.

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpacePermissionLevels`()

Get a list of possible space permission "levels" (ie. read, read/write, admin, etc.) and their capabilities. Permission levels are returned in order of increasing "magnitude". The reply of this call is constant and won't change until the agent is restarted.

Returns

```
[
        {
                "id": "none",
                "name: "None",
                "rights": {
                                "memberInvite": False,
                                "memberKick": False,
                                "memberSetRights": False,
                                "spaceRename": False,
                                "spaceDeleteOnServer": False,
                                "filePublish": False,
                                "fileDeleteOnServer": False,
                                "fileRename": False,
                                "fileRestoreFromTrash": False,
                                "fileChange": False
                                }
        },
        {
                "id": "read",
                "name": "Read"
                "rights": { ... }
        },
        ...
]
```

New in version 4.1.1.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpaceSettings`(*spaceId*)

Get space-specific settings.

New in version 4.1.2.

Parameters **`spaceId`** (*`int`*) – The Space id.

Return type dict

`getSpaceStatistics`(*id*)

Returns Space statistics

New in version 4.1.1.

See also:

`getSpace` (page 52), *`getSpaces`* (page 56) for other information.

Parameters **`id`** (*`int`*) – The Id of the Space

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getSpaces`(*sync=None*)

Returns a list of all known Spaces.

Note: As this function was added with TeamDrive 4, the fall back for TeamDrive 3 is to iterate over all Space Ids, therefore this function is much slower.

See also:

`getSpaceStatistics` (page 56) for statistics.

returns A list of *Space* (page 78) objects, for example

```
[
  {
        "name": "My Space",
        "creator": "$TMDR-1007",
        "icon": "archived",
        "status": "archived",
        "status_message": "Space Archived",
        "id": 1,
        "spaceRoot": "/home/teamdrive/Spaces/My Space",
        "time": "2015-09-15T11:17:48Z",
        "permissionLevel": "read"
  },
  {
        "name": "Api",
        "creator": "...",
        "spaceRoot": "/home/teamdrive/Spaces/Api",
        "icon": "space",
        "status": "active",
        "status_message": "OK",
        "id": 8,
        "fileId": 1,
        "time": "2015-09-15T11:17:48Z",
        "permissionLevel": "readWrite"
  }
]
```

New in version 4.0.

Parameters **`sync`** – (Optional, default `False`) If true, and the key repository has not been fetched from the server yet, wait for the key repository before returning

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getVersion`(*versionId*)

Returns a single version.

```
>>> versionId = 42
>>> TeamDriveApi().getVersion(versionId)["versionId"] == versionId
```

New in version 4.1.4.

Parameters **`versionId`** (*`int`*) – the id of the version

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`getVersions`(*fileId*)

Returns all versions of a file.

New in version 4.1.4.

Return type list[dict]

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`importSpaceKeys``()`

Import Space keys from the server (ie. Spaces that have been deleted locally will re-appear as inactive Spaces, if the key-repository backup was enabled)

Returns A dict where the key “spacesImported” contains the number of imported Spaces.

New in version 4.3.1.

Return type dict

`inviteMember``(`*spaceId*, *addressId*, *text*, *permissionLevel=None*, *password=None*, *name=None*, *sendEmail=True*`)`

Invites a user into a Space. If no addressId is given, and ‘name’ is set to the email address of someone who doesn’t have an account yet, they will get an email inviting them to join TeamDrive.

Using this function requires the HTTP POST method.

Throws *`TeamDriveException`* (page 76), if the call fails.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`addressId`** (*`int | None`*) – Already known Addressbook Id (optional if a name is provided)
- **`text`** (*`str`*) – Invitation Text
- **`name`** (*`str | None`*) – Username or an email address (required if no addressId is given)
- **`permissionLevel`** (*`str | None`*) – The new user permission Level. Optional.

  New in version 4.1.1.
- **`password`** (*`str | None`*) – Optional. The password for the invitation.

  New in version 4.1.1.
- **`sendEmail`** (*`bool | None`*) – Optional. If `False`, don’t email the invited user about the invitation (`True` by default).

  New in version 4.5.1.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Inviting that member failed
- **`ValueError`** – no JSON returned. Wrong URL?

`joinSpace``(`*id*, *disableFileSystem*, *spacePath=None*, *password=None*, *backupPath=None*, *fullRestore=None*, *snapshotId=None*, *sync=None*, *mountInVirtualFS=None*, *restoreForSnapshotExport=None*`)`

Accepts an invitation, joins an archived Space or rejoins an active one.

Note: Password protected Invitations: Invitations may be protected by a password and accepting these invitations require the password parameter.If an invitation is password protected, `getSpace` (page 52) adds `"isPasswordProtected": true` to the reply.

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails.

Parameters

- **`id`** (*`int`*) – The Id of the Space
- **`disableFileSystem`** (*`bool`*) – This will disable the synchronization into the file system. TeamDrive will only synchronize meta-data if this is *True*.
- **`spacePath`** (*`str | None`*) – Optional, Space root in the file system (must be absolute, to specify a foldername for the Space, use something like */users/bob/MySpace*, otherwise use */users/bob/*).
- **`password`** (*`str | None`*) – Optional, Password for accepting this invitation.

  New in version 4.1.1.
- **`backupPath`** (*`str`*) – An optional backup path in case the space already exists in the file system.

  New in version 4.2.1.
- **`fullRestore`** (*`bool`*) – Do a full restore of the space.

  New in version 4.2.1.
- **`snapshotId`** (*`int`*) – Do a preview-mode rollback of the Space to this snapshot (see `commitCurrentSnapshot` (page 33))

  New in version 4.3.2.
- **`restoreForSnapshotExport`** (*`bool`*) – If set to `True` along with a provided `snapshotId` parameter, the given snapshot will be placed in the Space folder on disc in paralell to the Space entering the “snapshot-preview” mode (see `commitCurrentSnapshot` (page 33))

  New in version 4.7.5.
- **`sync`** (*`bool`*) – Only return once the Space is active (this can take some time) (default is `False`)

  New in version 4.5.6.
- **`mountInVirtualFS`** (*`bool`*) – Whether to mount the space in the virtual file system.

  New in version 4.6.7.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Joining failed
- **`ValueError`** – no JSON returned. Wrong URL?

kickMember(*spaceId*, *addressId*)

Removes a member form this Space.

Removing a member from a Space will not remove this user from `getSpaceMemberIds` (page 54), but will set a this member to status "kicked" in `getMember` (page 50).

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails. For example, if the one doesn't have the rights to remove a user

New in version 4.0.

Parameters

- **spaceId** (*int*) – The Id of the Space
- **addressId** (*int*) – Already known Addressbook Id.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Removing this user failed
- **ValueError** – no JSON returned. Wrong URL?

lockFile(*fileId*)

Lock a file. Throws `TeamDriveException` (page 76), if the call fails.

New in version 4.2.1.

Parameters **fileId** (*int*) – The Id of the File

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

login()

A TeamDrive Client installation is bound to a single TeamDrive account. Calling `login` (page 60) authenticates TeamDrive user and creates a new device. this is typically the first call to an Agent.

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails.

Example as Command line:

```
./TeamDriveApi.py '127.0.0.1:45454' --user myUser --pass myPass login
```

Example with Curl:

```
curl 'https://127.0.0.1:45454/login' --data-binary '{"username":"My␣
↪User","password":"My Password"}'
```

Example:

```
>>> api = TeamDriveApi("127.0.0.1:45454", "My Username", "My Password")
>>> api.login()
```

Note: On the desktop client, this call is not available, as the API will not be available until the user has successfully logged in. This is a technical limitation of the Gui Client.

See also:

Chapter *Authentication* (page 27) regarding Authentication.

Parameters

- **username** (*str*) – The Username
- **password** (*str*) – The Password
- **tempPassword** (*str*) – A temporary password, in case of login from a lost-password procedure

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Logging failed.
- **ValueError** – no JSON returned. Wrong URL?

moveFile (*spaceId*, **args*)

Either *moveFile(self, spaceId, filePath, trashed, newFilePath)* or *moveFile(self, spaceId, fileId, newFilePath)* Moves a file from *filePath* or *fileId* to *newFilePath* in the given Space.

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails.

Parameters

- **spaceId** (*int*) – The Id of the Space
- **filePath** (*str*) – the path of the file in the Space.
- **trashed** (*int*) – Is the file in the trash?
- **fileId** – fileIf of the file meing moved
- **newFilePath** (*str*) – The destination file path.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.

- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Failed to queue command.
- **`ValueError`** – no JSON returned. Wrong URL?

`performExternalAuthentication`(*authToken*)

Login using an external authentication token. The registration server must be configured to validate the token, refer the chapter "External Authentication" in the "TeamDrive Registration Server Reference Guide" for details.

Parameters **`authToken`** (*`str`*) – The token generated by the authentication service

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Login failed.

`preLogin`(*username*)

Depending on configuration, some users may need to log in via an external authentication mechanism. This function can be used to check for any such settings before proceeding to login with the user.

Parameters **`username`** (*`str`*) – Username or email address

Return type dict

`publishFile`(*fileId*, *password=None*, *expiryDate=None*, *createShortUrl=None*, *encrypted=None*, *publishAs=None*, *versionId=None*)

Publishes the recent version of that file. The parameters *password* and *expiryDate* are optional. Throws *`TeamDriveException`* (page 76), if the call fails.

New in version 4.1.

Parameters

- **`fileId`** (*`int`*) – The Id of the File
- **`password`** (*`str`*) – If set, people need to enter this password to download the file
- **`expiryDate`** (*`str`*) – ISO 8601 extended format: either YYYY-MM-DD for dates or YYYY-MM-DDTHH:mm:ss, YYYY-MM-DDTHH:mm:ssTZD (e.g., 1997-07-16T19:20:30+01:00) for combined dates and times.
- **`createShortUrl`** (*`bool | None`*) – Whether to create a shortened URL

  New in version 4.6.3.
- **`encrypted`** (*`bool | None`*) – Enable server-side encryption of the published file

  New in version 4.6.3.
- **`publishAs`** (*`str`*) – Publish the file under a different name
- **`versionId`** (*`int`*) – Version ID, if publishing a specific file version

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.

- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Failed to queue command.
- **`ValueError`** – no JSON returned. Wrong URL?

`putFile` (*spaceId*, *path*, *data*)

This is a Python specific accessor to the "files" API. The files API is another API endpoint to handle file content instead of meta data. Urls used by the files API are generated by concatenating the API endpoint `files/` plus the local Space Id (eg. `3`) plus the full file path relative to the Space root (eg. `/my/path/to/the/file`). Such a url could look like this: `http://127.0.0.1:45454/files/3/my/path/to/the/file`. This API does not support reading and writing files that are in the trash.

This Uploads a file into a Space. The file creation is done asynchronously, a successful return code does not indicate a successful upload to the Server. Use *`getFiles`* (page 46) to request the current state of that file.

Example using telnet:

```
telnet 127.0.0.1 45454
PUT /files/3/testfile HTTP/1.1
Host: 127.0.0.1:45454
Accept-Encoding: identity
Content-Length: 4

test
```

Example using curl:

```
echo "test" | curl -X PUT -d @- http://127.0.0.1:45454/files/3/testfile
```

Returns

a dict containgng a *File* (page 79) object

```
{
  "file": {
        "confirmed": false,
        "creator": "local",
        "currentVersionId": 6156,
        "hasComments": false,
        "icon": "default",
        "id": 7195,
        "isDir": false,
        "name": "testfile",
        "path": "/",
        "permissions": "rw-rw-r--",
        "size": 4,
        "spaceId": 3,
        "time": "2015-10-22T14:09:45"
  },
  "result": true
}
```

Note: In TeamDrive 3, this API endpoint was called `/webdav`.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`path`** (*`str`*) – the path of the file in the Space.
- **`data`** – The binary data

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

`putFileContent`(*spaceId*, *spacePath*, *filePath*)

Note: This is not part of the HTTP API. See *`putFile`* (page 63) instead.

Example:

```
$ python ./TeamDriveApi.py '127.0.0.1:45454' putFileContent 3 /
↪testfile ./testfile
```

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`spacePath`** (*`str`*) – the path of the file in the Space.
- **`filePath`** (*`str`*) – the path to the local file

Return type dict

Raises **`httplib.HTTPException`** – a Connection Error.

`quit`(*logout*)

Quits TeamDrive.

Using this function requires the HTTP POST method.

Throws *`TeamDriveException`* (page 76), if the call fails.

Parameters **`logout`** (*`bool`*) – Quit and also log out the current user. See *`login`* (page 60) to log in afterwards.

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.

- **ValueError** – no JSON returned. Wrong URL?

removeComment(*commentId*)

Marks this comment as "removed"

Using this function requires the HTTP POST method.

See also:

getComments (page 45), *getSpaceComments* (page 53), *addComment* (page 31) and *addSpaceComment* (page 32)

New in version 4.1.1.

Parameters **commentId** (*int*) – The id of the comment that will be deleted.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Removing this comment failed
- **ValueError** – no JSON returned. Wrong URL?

removeLocallyFile(*id*, *recursive=False*)

Removes a file form the local file system by disabling the offline-available flag, but does not move this file into the trash or removes the file from the Space. This operation is not synchronized to other clients.

A successful return code does not indicate a successful execution, because it is done asynchronously.

Using this function requires the HTTP POST method.

See also:

deleteFileFromTrash (page 37) and *setTrashed* (page 72).

Parameters

- **id** (*int*) – the file to be removed
- **recursive** (*bool*) – Directories only: recursive remove

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

requestActivationEmail()

Request an activation-link email from the regserver for the locally logged in (but not activated) user.

Return type dict

requestResetPassword(*username*)

This call will reset the password of the user.

New in version 4.1.1.

Parameters **username** (*str*) – The Username

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Logging failed.
- **ValueError** – no JSON returned. Wrong URL?

resolveVersionConflict(*versionId*)

Changes the current version of a file to this version and resolves all version conflicts on this file.

Using this function requires the HTTP POST method.

Throws *TeamDriveException* (page 76), if the call fails.

See also:

setCurrentVersion (page 68)

New in version 4.1.4.

Parameters **versionId** (*int*) – The Id of the version

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

restoreLocallyFile(*id*, *recursive=False*)

Restores a file form the local file system. A successful return code does not indicate a successful execution, because it is done asynchronously. This operation is not synchronized to other clients.

Using this function requires the HTTP POST method.

Parameters

- **id** (*int*) – the file to be restored
- **recursive** (*bool*) – Directories only: recursive restore

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.

- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

sendSupportRequest (*contactName*, *contactEmail*, *contactTel*, *description*, *sendLogs*)

Uploads a support request to the server (see the `Client Log Files` chapter in the Registration server documentation on how to configure log uploads)

New in version 4.5.0.

Parameters

- **contactName** (*str | None*) –
- **contactEmail** (*str | None*) –
- **contactTel** (*str | None*) –
- **description** (*str | None*) – Description of the problem
- **sendLogs** (*bool | None*) – Whether or not to attach log files (true if not specified)

Return type dict

setAccountEmail (*email*)

Sets the account e-mail.

Using this function requires the HTTP POST method.

Returns True, if the license key was successfully changed otherwise raises *TeamDriveException* (page 76) Exception

See also:

setLicenseKey (page 68) to set the license key.

New in version 4.1.1.

Parameters **email** (*str*) – The new e-mail address

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?
- **TeamDriveException** – Failed to set email.

setAutoDelete (*fileId*, *autoDelete*)

In ‘Data Retention’ Spaces (see *Data Retention* (page 35)) with the “Auto Trash” option enabled, files are automatically moved to the trash after their retention period expires. This can be disabled per-file by setting `autoDelete` to `False` with this call.

New in version 4.6.3.

Parameters

- **fileId** (*int*) – The id of the file
- **autoDelete** (*bool*) – Whether to auto-trash the file

Return type dict

`setCurrentVersion`(*versionId*)

Changes the current version of a file to this version.

Using this function requires the HTTP POST method.

Throws *`TeamDriveException`* (page 76), if the call fails.

See also:

`getVersion` (page 57)

New in version 4.1.4.

Parameters **`versionId`** (*`int`*) – The Id of the version

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Failed to queue command.
- **`ValueError`** – no JSON returned. Wrong URL?

`setFeatureChoice`(*featureId*, *useNewFeature*, *spaceId=0*)

Set whether the given feature should be used in this Space (ie. suppresses further `Error_Feature_Upgrade` responses for this Space and feature)

See also:

Chapter *`Error_Feature_Upgrade`* (page 77)

Note that a response with `useNewFeature == False` will only suppress further error responses for features that have an older behaviour to fall back to.

New in version 4.6.3.

Parameters

- **`featureId`** (*`int`*) – The id of the feature
- **`useNewFeature`** (*`bool`*) – Whether to use the new feature
- **`spaceId`** (*`int | None`*) – (Optional, depending on the feature) The Space that this choice is being made for

Return type dict

`setLastViewTime`(*id*)

Marks the comments for the given Space as ‘read’.

Parameters **`spaceId`** (*`int`*) – Id of the Space

Return type dict

`setLicenseKey`(*licenseKey*)

Overwrite the license key of this agent.

Using this function requires the HTTP POST method.

Returns True, if the license key was successfully changed otherwise raises *`TeamDriveException`* (page 76) Exception

See also:

`setAccountEmail` (page 67) to set the registration e-mail address. `getLicense` (page 49) to retrieve license infos.

Parameters **`licenseKey`** (*`str`*) – The new license key

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`setLocalEncryptionEnabled`(*enabled*)

Enable/disable the encryption of local files.

Parameters **`enabled`** (*`bool`*) – Whether or not to enable the encryption of all local files.

Return type dict

`setMemberPermissionLevel`(*spaceId*, *addressId*, *permissionLevel*)

Changes the permission level of another member.

Using this function requires the HTTP POST method.

Throws `TeamDriveException` (page 76), if the call fails. For example, if the one doesn't have the rights to remove a user

New in version 4.1.1.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`addressId`** (*`int`*) – Already known Addressbook Id.
- **`permissionLevel`** (*`str`*) – the new permission level

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`TeamDriveException`** – Removing this user failed
- **`ValueError`** – no JSON returned. Wrong URL?

`setProfileInfo`(*email=None*, *phone=None*, *mobile=None*, *fullName=None*, *miscellaneous=None*)

Sets profile information of the current user.

See also:

`setProfilePicture` (page 70) to set the profile picture.

New in version 4.1.1.

Parameters

- **email** (*str | None*) – Optional, the new e-mail.
- **phone** (*str | None*) – Optional, the new phone number.
- **mobile** (*str | None*) – Optional, the new mobile number.

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

setProfilePicture (*data*)

Sets the profile picture of the current user.

See also:

getProfilePicture (page 50) to retrieve profile pictures and *clearProfilePicture* (page 33) to clear the picture.

New in version 4.1.1.

Parameters **data** (*bytes*) – The blob.

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

setProxyCredentials (*username*, *password*)

Set authentication credentials for the configured proxy server.

:param username :type username: str :param password :type password: str :rtype: dict

setProxySettings (*type*, *host*, *port*, *username*, *pasword*, *pacFile*)

Configure a proxy service for TeamDrive.

Parameters

- **type** (*str*) – One of `none`, `system`, `manual`, or `automatic`
- **host** (*str*) – Proxy hostname
- **port** (*int*) – Port
- **username** (*str*) –
- **pasword** (*str*) –
- **pacFile** (*str*) – Proxy Configuration URI

Return type dict

setRetentionStartTime (*fileId*, *startTime*)

In ‘Data Retention’ Spaces (see *Data Retention* (page 35)), files are supposed to be un-deletable for a given minimum period of time (the “retention period”). In some cases, however, the actual legal retention period for a file may already have begun before the Space was created -> in such case, this API call can be used to indicate the date at which the retention period for this file actually started.

New in version 4.6.3.

Parameters

- **`fileId`** (*`int`*) – The id of the file
- **`startTime`** (*`str`*) – Date from which the retention period for the file should begin (YYYY-MM-DD)

Return type dict

`setSetting` (*key*, *value*)

Set a Setting. must be a valid key as in *`getSettings`* (page 52). An incomplete list of settings can be found in the *Options* (page 21) Section.

Using this function requires the HTTP POST method.

Parameters

- **`key`** (*`str`*) – The name of the setting
- **`value`** – The new value of the setting

Return type dict

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`setSpacePath` (*id*, *path*)

Set the root path for a Space in the local file system. This can be used to move a Space to a new location, or to resolve a "Space root not found error" (eg. when a local folder name has changed and TeamDrive can no longer find the Space).

Note: Because this function affects the file system, it can only be called from the host that the API is running on (calling it from elsewhere will result in an "Unknown function" error).

Using this function requires the HTTP POST method.

New in version 4.5.0.

Parameters

- **`spaceId`** (*`int`*) – The Space id.
- **`localPath`** (*`str`*) – New root folder for the Space

Return type dict

`setSpaceSetting` (*spaceId*, *key*, *value*)

set a Space Setting. must be a valid key as in *`getSpaceSettings`* (page 56).

Using this function requires the HTTP POST method.

New in version 4.1.2.

Parameters

- **`spaceId`** (*`int`*) – The Space id.

- **key** (*str*) – The name of the setting
- **value** – The new value of the setting

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **ValueError** – no JSON returned. Wrong URL?

setSpaceSyncEnabled(*id*, *enabled*)

Enable/disable synchronization for a Space

Using this function requires the HTTP POST method.

New in version 4.5.0.

Parameters

- **spaceId** (*int*) – The Space id.
- **enabled** (*bool*) – Whether to enable synchronization

Return type dict

setTrashed(*fileId*, *inTrash*)

Moves a file into or out of the trash. Trashing a file also removes the file from the local file system.

Throws *TeamDriveException* (page 76), if the call fails.

Call *deleteFileFromTrash* (page 37) to delete the file physically afterwards.

See also:

removeLocallyFile (page 65) and *deleteFileFromTrash* (page 37).

New in version 4.0.

Parameters

- **fileId** (*int*) – The Id of the Space
- **inTrash** (*bool*) – Is the file in the trash?

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

sse()

This is the Server Side Events api listening on */sse*. This API sends events according to the Server Side Events specification.

The implementation of the python wrapper requires Python 3.

Example Usage:

```
$ python3 TeamDriveApi.py '127.0.0.1:45454' sse
data: { "event" : "FileEntryChanged", "fileId" : 0, "hasConflict" : 0, "isDir
↪" :
                false, "isTrashed" : false, "name" : "my file", "path" :
                "/home/teamdrive/Spaces/My Space", "spaceId" : 3, "synced" :␣
↪true }

data: { "event" : "SpaceModified", "spaceId" : 3 }
```

Possible Event types are: *FileEntryChanged*, *VersionEntryChanged*, *FileVersionPublished*, *FileVersionUnpublished* and *SpaceModified*

Raises

- **`httplib.HTTPException`** – a Connection Error.
- **`socket.error`** – a Connection Error.
- **`ConnectionRefusedError`** – a Connection Error.
- **`ValueError`** – no JSON returned. Wrong URL?

`start2FA`(*supportedTypes='email-otp'*)

Start the two-factor authentication process. The `supportedTypes` parameter tells the server which 2FA-schemes are supported, and the value of `authType` in the response indicates which method is being used (which method the server selects depends on which methods the server supports, as well as server/account/user settings).

Possible 2FA methods (at time of writing):

- **`email-otp`: The user gets an email with a password that can be passed to a server (via check2FA(<paswd>, ‘er**
  a link that can alternatively be clicked to complete 2FA (this can then be polled for by calling check2FA(‘’,’email-otp’)). The email can be re-sent by calling start2FA() again.
- \``goog-auth`\`: 2FA via the “Google Authenticator” app. The password can be submitted to the server via check2FA(<passwd>, ‘goog-auth’).

The API indicates that 2fa is required by returning error code 58 in response to the :any:**‘**login’ call.

Parameters **`supportedTypes`** (*`str`*) – A comma-separated list of 2FA-methods supported by the client, eg. “goog-auth,email-otp”. Defaults to “email-otp”.

Return type dict

`tempPasswordLogin`(*username*, *tempPassword*, *password*, *verifyPassword*, *pin=None*)

Login from a “forgot password” procedure using a temporary password, changing the account password in the process,

Parameters

- **`username`** (*`str`*) –
- **`tempPassword`** (*`str`*) – The temporary password
- **`pasword`** (*`str`*) – The new account password
- **`verifyPassword`** (*`str`*) – Repeat of the new account password
- **`pin`** (*`str`*) – If enabled, an existing installation can be logged in to by providing the local pin instead of the account password.

Return type dict

unlockFile (*fileId*)

Lock a file. Throws *TeamDriveException* (page 76), if the call fails.

New in version 4.2.1.

Parameters **fileId** (*int*) – The Id of the File

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

unpublishFile (*fileId*, *versionId=None*)

Unpublishes the file. Throws *TeamDriveException* (page 76), if the call fails.

New in version 4.1.

Parameters

- **fileId** (*int*) – The Id of the File
- **versionId** (*int*) – Version ID, if unpublishing a specific file version

Return type dict

Raises

- **httplib.HTTPException** – a Connection Error.
- **socket.error** – a Connection Error.
- **ConnectionRefusedError** – a Connection Error.
- **TeamDriveException** – Failed to queue command.
- **ValueError** – no JSON returned. Wrong URL?

unsetFeatureChoice (*featureId*, *spaceId*)

Cause `Error_Feature_Upgrade` responses for this Space and feature to be shown even if they were suppressed by a previous `setFeatureChoice` call.

See also:

Chapter *Error_Feature_Upgrade* (page 77)

New in version 4.6.3.

Parameters

- **featureId** (*int*) – The id of the feature
- **spaceId** (*int | None*) – (Optional, depending on the feature) The Space that this choice is being made for

Return type dict

updateInbox (*id*, *expiryTime=None*, *password=None*, *description=None*, *maxFileCount=None*, *maxFileCountInterval=None*, *maxFileSize=None*, *maxSize=None*)

Update the configuration of an existing inbox.

Parameters

- **id** (*int*) – The id of the Inbox
- **expiryTime** – See *createInbox* (page 35)
- **password** – See *createInbox* (page 35)
- **description** – See *createInbox* (page 35)
- **maxFileCount** – See *createInbox* (page 35)
- **maxFileCountInterval** – See *createInbox* (page 35)
- **maxFileSize** – See *createInbox* (page 35)
- **maxSize** – See *createInbox* (page 35)

Return type dict

zipCancel (*spaceId*, *path*)

Cancel/clean up a zip-folder task, see *zipFolder* (page 75)

Note: This is not an API function, just a python wrapper for the 'zip' endpoint, see *zipFolder* (page 75)

New in version 4.5.0.

Parameters

- **spaceId** (*int*) – The Id of the Space
- **path** (*str*) – the path of the folder in the Space.

Return type dict

zipDownload (*spaceId*, *path*)

Download the result archive of a zip-folder task, see *zipFolder* (page 75)

Note: This is not an API function, just a python wrapper for the 'zip' endpoint, see *zipFolder* (page 75)

New in version 4.5.0.

Parameters

- **spaceId** (*int*) – The Id of the Space
- **path** (*str*) – the path of the folder in the Space.

zipFolder (*spaceId*, *path*, *recursive*)

This is a Python specific accessor to the "zip" API which can be used to download a .zip archive of a folder. Because downloading, zipping, and returning the archive can take a long time depending on the size of the folder, this process is broken up into several steps:

1. **POST to `<api endpoint>/zip[Recursive]/<space id><folder path>`** This will start the process of downloading the neccessary files and zipping them. As indicated above, use `.../zipRecursive/...` to include subfolders in the archive, or `.../zip/...` to only include the top level
2. **GET to `<api endpoint>/zipProgress/<space id><folder path>`** Use this request to poll the status of the task. Returns a json object like:

```
{
        "totalFiles": 100,
        "filesZipped": 50,
        "status": "status text",
        "isDone": false
}
```

3. **GET** **`<api endpoint>/zip/<space id><folder path>`** Downloads the finished archive
4. **DELETE** **`<api endpoint>/zip/<space id><folder path>`** This will clean up resources for this job on the server, and cancel the job if is still running.

New in version 4.5.0.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`path`** (*`str`*) – the path of the folder in the Space.
- **`recursive`** (*`bool`*) – whether to recursively zip subfolders

Return type dict

`zipProgress`(*spaceId*, *path*)

Get the status of a running zip-folder task, see *`zipFolder`* (page 75)

Note: This is not an API function, just a python wrapper for the 'zip' endpoint, see *`zipFolder`* (page 75)

New in version 4.5.0.

Parameters

- **`spaceId`** (*`int`*) – The Id of the Space
- **`path`** (*`str`*) – the path of the folder in the Space.

Return type dict

11.6.2 TeamDriveApi.TeamDriveException

class `TeamDriveApi.`**`TeamDriveException`**(*name*, *result*)

In case of an error, the TeamDrive API returns a specific JSON document. Thy python Module will parse this error.

Example using curl:

```
curl -s '127.0.0.1:45454/api/getSpaces'
{
  "error": 50,
  "error_message": "CJsonApi::ApiError (Error_Missing_Session)",
  "result": false,
  "status_code": 500
}
```

Example using the command line:

```
$ ./TeamDriveApi.py '127.0.0.1:45454' getSpaces
Traceback (most recent call last):
  File "./TeamDriveApi.py", line 1178, in <module>
        main()
  File "./TeamDriveApi.py", line 1174, in main
        result = (getattr(TeamDriveApi, command)(api, *params))
  File "./TeamDriveApi.py", line 307, in getSpaces
        return self._call("getSpaces")
  File "./TeamDriveApi.py", line 204, in _call
        return self._call(name, params, method)
  File "./TeamDriveApi.py", line 205, in _call
        raise TeamDriveException(name, jsonRes)
__main__.TeamDriveException: Call getSpaces failed with {u'status_code': 500, u
↪'error_message': u'CJsonApi::ApiError (Error_Missing_Session)', u'result':␣
↪False, u'error': 50}
```

`Error_Authentication_Required` = 55

This code is returned if a function requires that the user have entered his password for the current API session. An example where this might be returned is when the user has a persistent login to a local installation - if this code is returned, the user must re-login to authenticate himself before performing the API call which returned this code a second time.

`Error_Feature_Upgrade` = 70

The API call would force other members of the Space to upgrade their TeamDrive version, a confirmation step is required via `setFeatureChoice`.

In addition to the usual fields, the returned object will include the following:

```
{
  ...
  "featureId": 10,
  "featureSpaceId": 123,
  "featureQuestion": "Using XYZ new feature will require other Space members␣
↪to upgrade their installation, continue?",
  "featureHasOldBehaviour": True,
  ...
}
```

- `featureId` is the ID of the feature
- `spaceId` is the ID of the Space that this feature-warning applies to (may be omitted for a feature that is not space-specific)
- `featureQuestion` is a prompt that could be displayed to a user
- `featureHasOldBehaviour` -> if `True`, there is an older behaviour to fall back to, and calling `setFeatureChoice` with `False` will simply cause further API calls to use the old behaviour

See also:

`setFeatureChoice` (page 68).

Note: This status is only returned if the setting `http-api-enable-feature-alerts` is `True` (the default is `False`)

`Error_HTTP_Method_Not_Allowed` = 2

`Error_Insufficient_Rights` = 30

`Error_Internal_Error` = 1

`Error_Invalid_File_Key` = 43

`Error_Invalid_Parameter` = 21

`Error_Missing_Parameter` = 20

`Error_Missing_Session` = 50

No or invalid credentials supplied.

See also:

Chapter *Authentication* (page 27).

`Error_Unknown_AddressId` = 42

`Error_Unknown_Error` = 3

`Error_Unknown_Space` = 40

`Error_Unknown_User` = 41

`Error_WebportalAccessDisabled` = 33

Returned when an attempt is made to access a space from the webportal, when that space has webportal-access disabled

`Error_Wrong_Login` = 51

`getError``()`

The TeamDrive API Exception code :rtype: int

`getErrorString``()`

A human readable version of this error :rtype: str

`getStatusCode``()`

The HTTP status code of this exception. :rtype: int

11.7 Common JSON Data Types

The API returns common data types encoded in JSON objects for different API functions. This section provides an overview of some of these data types.

11.7.1 Space

This JSON object describes meta information about a single Space.

Member	Description
`id`	Id of the Space. Used to identify a specific Space
`fileId`	FileId of the root-folder in this Space. Only present for active Spaces
`name`	Name of the Space
`spaceRoot`	Local path to the Space.
`invitor`	Invitations only: User name who created the invitation.
`creator`	User name who created the Space
`icon`	Space icon.
`status`	Status of the Space.
`status_message`	Describes the current status of this Space.
`hasTrashedFiles`	Has trashed files in the Space.
`isPasswordProtected`	Invitations only: This invitation needs a password. Obmitted if false.
`time`	Datetime of the last modification of this Space.
`text`	Invitation comment
`permissionLevel`	Permission level of the current user in this Space. (Not for invitations)
`hasComments`	Space has comments. (Not for invitations)
`newComments`	Amount of new comments in the Space. (Not for invitations)
`isOfflineAvailable`	“true”, if the Space will be synchronized to the local file system. TeamDrive will only synchronize meta-data if this is *false* (Not for invitations)

Example:

```
{
        "name": "Api",
        "creator": "...",
        "spaceRoot": "/home/teamdrive/Spaces/Api",
        "icon": "space",
        "status": "active",
        "status_message": "OK",
        "id": 8,
        "fileId": 1
}
```

11.7.2 File

This JSON object describes meta information about a single File or diretory.

Member	Description
`name`	File or Folder Name
`id`	Local Id. Mainly used as parameter.
`spaceId`	Space Id of the file.
`path`	Path to the file excluding the name.
`currentVersionId`	Version id that should be in the file system
`isDir`	Boolean value. `true` for directories.
`hasNameConflict`	File or Folder has a name conflict
`hasVersionConflict`	File has a version conflict
`permissions`	Unix permissions
`creator`	User name of the file creator.
`created`	file creation time. (Files only)
`icon`	icon name based on the file extension. (Files only)
`confirmed`	The current version is synchronized to the server. (Files only)
`size`	The size in bytes of the current version. (Files only)
`time`	file creation time. (Files only)
`lastModified`	Last modification time. (Files only)
`publicUrl`	URL to the file if it is published. (Files only)
`hasComments`	Boolean value. (Files only)
`isLockedByMe`	Boolean value. (Files only)
`isLockedByOther`	Boolean value. (Files only)
`trasher`	User name who deleted the file. (Files only)
`trashed`	deletion time. (Files only)

Example:

```
{
    "isDir": true,
    "name": "My folder",
    "currentVersionId": 0,
    "spaceId": 3,
    "isInFileSystem": true,
    "path": "/",
    "id": 756,
    "permissions": "rwxr-xr-x"
}
```

11.7.3 Addressbook Entry

This JSON object describes meta information about a addressbook entry.

Member	Description
`id`	Addressbook entry Id
`name`	User display name
`email`	Registration Email
`profile`	Profile information
`initials`	User’s initials

Where Profile information is a JSON object containing:

Member	Description
`email`	Display Email
`phone`	Phone number
`mobile`	Mobile nuber
`hasProfilePicture`	Boolean value. `true` for User’s with profile picture.

CHAPTER

TWELVE

CHANGE LOGS

12.1 TeamDrive Agent 4.6.3

api:

- added setting http-api-enable-feature-alerts (default is False)
- if http-api-enable-feature-alerts is set to True, any operation that would force at least one other member of the Space to upgrade their installation will result in the API response *Error_Feature_Upgrade* (page 77). The new API call *setFeatureChoice* (page 68) then needs to be used to explicitly enable or disable the new functionality for the Space
- added new API calls *setFeatureChoice* (page 68) and *unsetFeatureChoice* (page 74)
- new parameters for *createSpace* (page 35): dataRetentionPeriod, dataRetentionAutoTrash
- new parameters for *publishFile* (page 62): createShortUrl, encrypted
- new API call *setAutoDelete* (page 67)
- new API call *setRetentionStartTime* (page 70)

web interface:

- confirmation dialog for actions that would require other Space members to update TeamDrive
- create and configure "Data Retention" Spaces (see *Data Retention* (page 35))

12.2 TeamDrive Agent 4.5.6

api:

- added a sync parameter to the *joinSpace* (page 58) API function which causes it to return only after the Space has become active
- added a sync parameter to the *getSpaces* (page 56) API function: if the key repository has not been fetched from the server yet, the function will wait for the server to be queried before returning
- the sortBy parameter for the function *findFiles* (page 39) can now be set to added to sort the returned files by the time they were added to the Space
- added support for Space setting fsLockingEnabled
- added setting enable-fs-locking

web interface:

- toggle file system write protection of soft-locked files with the global setting `enable-fs-locking` and the per-Space setting `fsLockingEnabled` (non-webportal GUI only)

12.3 TeamDrive Agent 4.5.3

web interface:

- added option to delete all trashed files in a Space ('empty trash')

api:

- added API function *`emptyTrash`* (page 39)

12.4 TeamDrive Agent 4.5.2

Minor bugfixes. See Forum releas notes https://forum.teamdrive.net/viewtopic.php?f=2&t=3182

web interface:

- added setting to enable / disable synchronization globally (non-webportal GUI only)
- toggle Space offline-availability (non-webportal GUI only)

12.5 TeamDrive Agent 4.5.1

web interface:

- fixed bug where the 'unpublish all versions' button was only available if the current version was published
- paging in long lists of files (paging is implemented by loading more data as the page is scrolled)
- natural string sort in list of files when sorting by file name
- added an advanced file search within spaces, with options to search subfolders and filter by various status properties such as available-offline, conflicted, published, etc.
- added a global file search
- view and configure Read Confirmations

api:

- added API function *`findFiles`* (page 39) for advanced file searching, with paged results
- files that have been restored from the trash no longer have the `trashed` and `trasher` fields set
- added API function *`getReadConfirmations`* (page 50)
- fixed crash caused by interrupted file download requests
- fixed a bug related to multiple paralell download requests for the same file
- added `sendEmail` parameter to the *`inviteMember`* (page 58) function

12.6 TeamDrive Agent 4.5.0

web interface:

- download folders as zip archive
- fixed a bug displaying Spaces with synchronization switched off
- fixed incorrect folder size information on the folder info page
- added support-request form to the profile info page
- the Space info page now includes a count of locally-available files (non-webportal GUI only)
- the Space info displays the amount of space taken up by server snapshots
- display size of individual snapshots in the snapshot list
- can now delete snapshots

api:

- added an endpoint to download folders as zip archives, see `zipFolder` (page 75)
- added an API function to enable/disable Space synchronization, see `setSpaceSyncEnabled` (page 72)
- added API function `getFolderInfo` (page 47)
- added API function `sendSupportRequest` (page 67)
- added API function `setSpacePath` (page 71)
- added API function `deleteSnapshot` (page 37)

12.7 TeamDrive Agent 4.3.3

web interface:

- added preview screen for images, text files and pdfs (only applies when not running on localhost)
- added trashExpiry setting to the "Space Overview" page

api:

- can get/set 'trashExpiry' for a Space using `getSpaceSettings` (page 56) and `setSpaceSetting` (page 71)

12.8 TeamDrive Agent 4.3.2

web interface:

- fixed a bug uploading files
- fixed a bug where files would get downloaded with the wrong name
- added filter options to the list of activities
- can now import space keys from the server
- added button to refresh the Space list
- added button to hide inactive Spaces

- can now open or delete previous versions of a file
- can now open trashed files
- added more sorting options to the list of Spaces and lists of files
- fixed files with an unknown extension getting uploaded with an empty content-type header
- added handling for point-in-time recovery / snapshots

API:

- added filter parameters to the getActivities() function
- fixed a typo in the 'expiryDate' parameter for the publishFile() function (the incorrect 'expiraryDate' is now deprecated)
- added importSpaceKeys() function
- added getSnapshots(), createSnapshot(), and commitCurrentSnapshot() functions

12.9 TeamDrive Agent 4.3.1

web interface:

- if the webinterface is being used with a local installation, the files are accessed directly in the filesystem
- return to the last opened page after logging back in due to a session timeout
- indicate when a file has published versions
- fixed urls for folders with special characters
- display file errors (eg. invalid path/name)
- can now import and export backups
- added restore-Space function

API:

- fixed a bug handling list parameters
- fixed a bug uploading new versions of the same file
- added functions to import and export backups

12.10 TeamDrive Agent 4.2.1

web interface:

- added file version page
- name- and version-conflicts are highlighted and can be resolved using the web interface
- fixed a bug uploading files on internet explorer
- added advanced options when joining or activating a space (offline availability, space path, restore from backup, full restore)
- added move-space function to the space info page
- added a 'notify Space members by Email' option when creating Space or file comments

12.11 TeamDrive Agent 4.2.0

- changed: `getActivities` (page 41), `getActivityDetails` (page 43).

12.12 TeamDrive Agent 4.1.4

- added `getVersion` (page 57), `getVersions` (page 57), `deleteVersion` (page 38), `resolveVersionConflict` (page 66) and `setCurrentVersion` (page 68).

12.13 TeamDrive Agent 4.1.3

- deprecated: `teamdrived --language` (page 21).

12.14 TeamDrive Agent 4.1.2

- added `setSpaceSetting` (page 71).
- added `teamdrived --enable-shell-extension` (page 22).

12.15 TeamDrive Agent 4.1.1

- added `setAccountEmail` (page 67)
- `about` (page 31) no longer returns license info. call `getLicense` (page 49) instead.
- added `requestResetPassword` (page 65)
- added `getProfilePicture` (page 50), `setProfilePicture` (page 70), `clearProfilePicture` (page 33) and `setProfileInfo` (page 69)
- added `getComments` (page 45), `getSpaceComments` (page 53), `addComment` (page 31), `addSpaceComment` (page 32) and `removeComment` (page 65)
- deprecated `getSpaceMemberIds` (page 54)
- added `getSpacePermissionLevels` (page 55), `setMemberPermissionLevel` (page 69), added permission levels to Member related calls.
- added `getSpaceStatistics` (page 56)
- added ability to handle password protected invitations in `joinSpace` (page 58) and `inviteMember` (page 58)
- added `getActivities` (page 41) and `getActivityDetails` (page 43)
- added `teamdrived --idle-shutdown-timeout` (page 24), `teamdrived --http-api-external-register-url` (page 24) and `teamdrived --http-api-external-login-url` (page 24)

12.16 TeamDrive Agent 4.1

- Added `createFolder` (page 34), `publishFile` (page 62) and `unpublishFile` (page 74).
- Added documentation of the */sse* API

12.17 TeamDrive Agent 4.0

- All modifying calls require HTTP POST requests, for example createFolder or deleteSpace
- Client Authentication is **required**
- Added `getSpaces` (page 56), `getFullFolderContent` (page 48), `getFullAddressbook` (page 47), `setTrashed` (page 72) and `createFolder` (page 34)
- Changed the Exception response.
- added `getMember` (page 50) to retrieve the status of a member

12.18 TeamDrive Agent 3.3.1

- added `getSpaceMemberIds` (page 54)

12.19 Web User Interface 1.0.4

- Added Space activities

12.20 Web User Interface 1.0.2

- Create Space Dialog: Added Server selection. (Requires Agent 4.1.2).
- Improved error message if there is no depot usage available.
- Added automatic redirect to the external login url on the login page.
- Fixed a bug regarding custom user rights (Only TeamDrive 3 Spaces affected)

CHAPTER THIRTEEN

INDICES AND TABLES

- genindex