Nextcloud Client Manual

Release 3.14.1

The Nextcloud developers

Oct 26, 2024

CONTENTS

CHAPTER

ONE

INTRODUCTION

Available for Windows, macOS, and various Linux distributions, the Nextcloud Desktop Sync client enables you to:

- Specify one or more directories on your computer that you want to synchronize to the Nextcloud server.
- Always have the latest files synchronized, wherever they are located.

Your files are always automatically synchronized between your Nextcloud server and local PC.

1.1 Improvements and New Features

The 3.14 release of the Nextcloud desktop sync client has many new features and improvements.

- Main dialog will be a regular window if tray icons are not available on the system.
- Virtual files will be optional. That enables the desktop client to run on older Windows versions.
- Improvements for the virtual files on Windows
- Network traffic performance improvements
- Improvements to the sync engine

CHAPTER

TWO

INSTALLING THE DESKTOP SYNCHRONIZATION CLIENT

You can download the latest version of the Nextcloud Desktop Synchronization Client from the Nextcloud download page. There are clients for Linux, macOs, and Microsoft Windows.

The currently supported server releases are the latest three stable versions at time of publication. It means that the 3.14 release series is supporting stable server major versions. See https://github.com/nextcloud/server/wiki/Maintenance-and-Release-Schedule for supported major versions.

Installation on Mac OS X and Windows is the same as for any software application: download the program and then double-click it to launch the installation, and then follow the installation wizard. After it is installed and configured the sync client will automatically keep itself updated; see *The Automatic Updater* for more information.

Linux users must follow the instructions on the download page to add the appropriate repository for their Linux distribution, install the signing key, and then use their package managers to install the desktop sync client. Linux users will also update their sync clients via package manager, and the client will display a notification when an update is available.

Linux users must also have a password manager enabled, such as GNOME Keyring or KWallet, so that the sync client can login automatically.

You will also find links to source code archives and older versions on the download page.

2.1 System Requirements

- Windows 10+ (64-bits only)
- macOS 11.4+ (64-bits only)
- Linux (ubuntu 22.04 or openSUSE 15.5 or . . .) (64-bits only)

Note: For Linux distributions, we support, if technically feasible, the current LTS releases. For BSD, we support them if technically feasible but we do not test

2.2 Customizing the Windows Installation

If you just want to install Nextcloud Desktop Synchronization Client on your local system, you can simply launch the *.msi* file and configure it in the wizard that pops up.

2.2.1 Features

The MSI installer provides several features that can be installed or removed individually, which you can also control via command-line, if you are automating the installation, then run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi
```

The command will install the Nextcloud Desktop Synchronization Client into the default location with the default features enabled. If you want to disable, e.g., desktop shortcut icons you can simply change the above command to the following:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi REMOVE=DesktopShortcut
```

See the following table for a list of available features:

Feature	Enabled by default	Description	Property to disable
Client	Yes, required	The actual client	
DesktopShortcut	Yes	Adds a shortcut to the desktop	`NO_DESKTOP_SHORTCUT`
StartMenuShortcuts	Yes	Adds a shortcut to the start menu	`NO_START_MENU_SHORTCUTS`
ShellExtensions	Yes	Adds Explorer integration	`NO_SHELL_EXTENSIONS`

Installation

You can also choose to only install the client itself by using the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT=Client
```

If you for instance want to install everything but the `DesktopShortcut` and the `ShellExtensions` feature, you have two possibilities:

1. You explicitly name all the features you actually want to install (whitelist) where *Client* is always installed anyway:

   ```
   msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT=StartMenuShortcuts
   ```

2. You pass the *NO_DESKTOP_SHORTCUT* and *NO_SHELL_EXTENSIONS* properties:

   ```
   msiexec /passive /i Nextcloud-x.y.z-x64.msi NO_DESKTOP_SHORTCUT="1" NO_SHELL_
   ↪EXTENSIONS="1"
   ```

Note: The Nextcloud *.msi* remembers these properties, so you don't need to specify them on upgrades.

Note: You cannot use these to change the installed features, if you want to do that, see the next section.

Changing Installed Features

You can change the installed features later by using *REMOVE* and *ADDDEFAULT* properties.

1. If you want to add the the desktop shortcut later, run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi ADDDEFAULT="DesktopShortcut"
```

2. If you want to remove it, simply run the following command:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi REMOVE="DesktopShortcut"
```

Windows keeps track of the installed features and using *REMOVE* or *ADDDEFAULT* will only affect the mentioned features.

Compare REMOVE and ADDDEFAULT on the Windows Installer Guide.

Note: You cannot specify *REMOVE* on initial installation as it will disable all features.

2.2.2 Installation Folder

You can adjust the installation folder by specifying the *INSTALLDIR* property like this:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi INSTALLDIR="C:\Program Files\Non Standard␣
↪Nextcloud Client Folder"
```

Be careful when using PowerShell instead of *cmd.exe*, it can be tricky to get the whitespace escaping right there. Specifying the *INSTALLDIR* like this only works on first installation, you cannot simply re-invoke the *.msi* with a different path. If you still need to change it, uninstall it first and reinstall it with the new path.

2.2.3 Disabling Automatic Updates

To disable automatic updates, you can pass the *SKIPAUTOUPDATE* property.:

```
msiexec /passive /i Nextcloud-x.y.z-x64.msi SKIPAUTOUPDATE="1"
```

2.2.4 Launch After Installation

To launch the client automatically after installation, you can pass the *LAUNCH* property.:

```
msiexec /i Nextcloud-x.y.z-x64.msi LAUNCH="1"
```

This option also removes the checkbox to let users decide if they want to launch the client for non passive/quiet mode.

Note: This option does not have any effect without GUI.

2.2.5 No Reboot After Installation

The Nextcloud Client schedules a reboot after installation to make sure the Explorer extension is correctly (un)loaded. If you're taking care of the reboot yourself, you can set the *REBOOT* property:

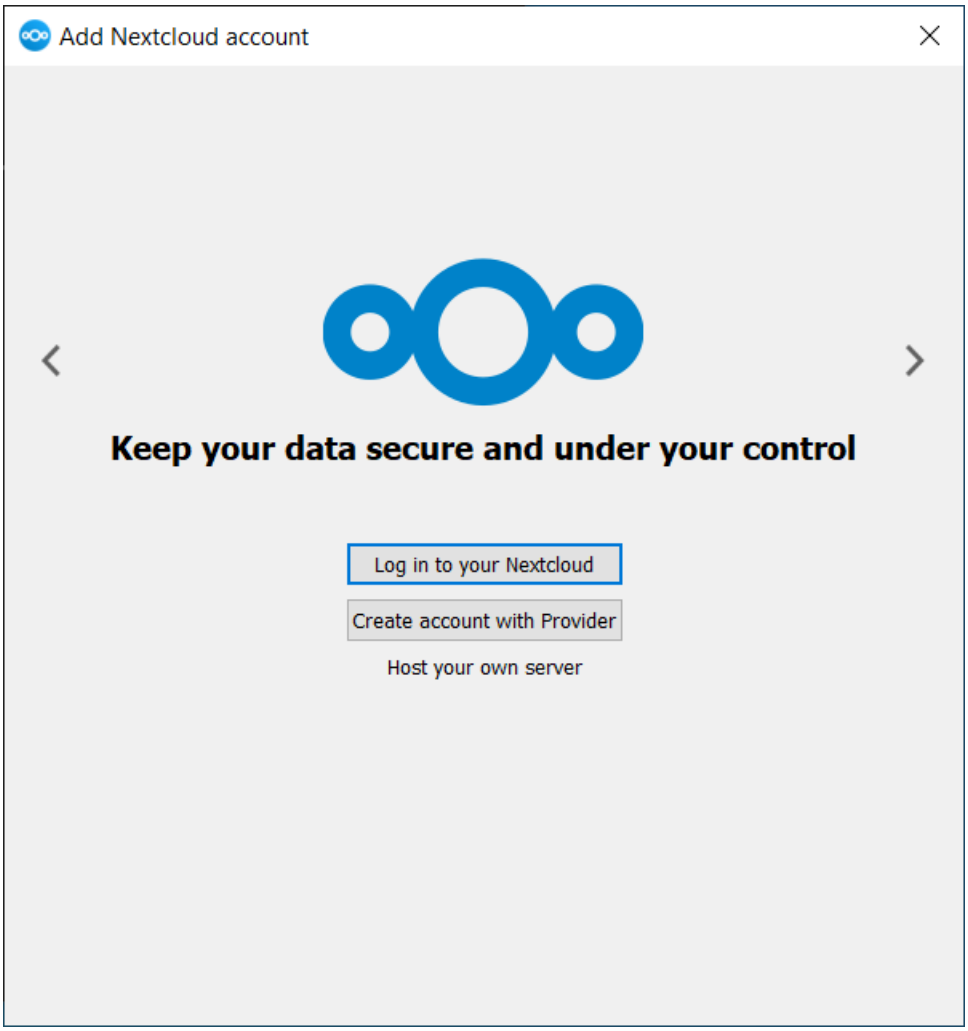
```
msiexec /i Nextcloud-x.y.z-x64.msi REBOOT=ReallySuppress
```

This will make *msiexec* exit with error *ERROR_SUCCESS_REBOOT_REQUIRED* (3010). If your deployment tooling interprets this as an actual error and you want to avoid that, you may want to set the *DO_NOT_SCHEDULE_REBOOT* instead:

```
msiexec /i Nextcloud-x.y.z-x64.msi DO_NOT_SCHEDULE_REBOOT="1"
```

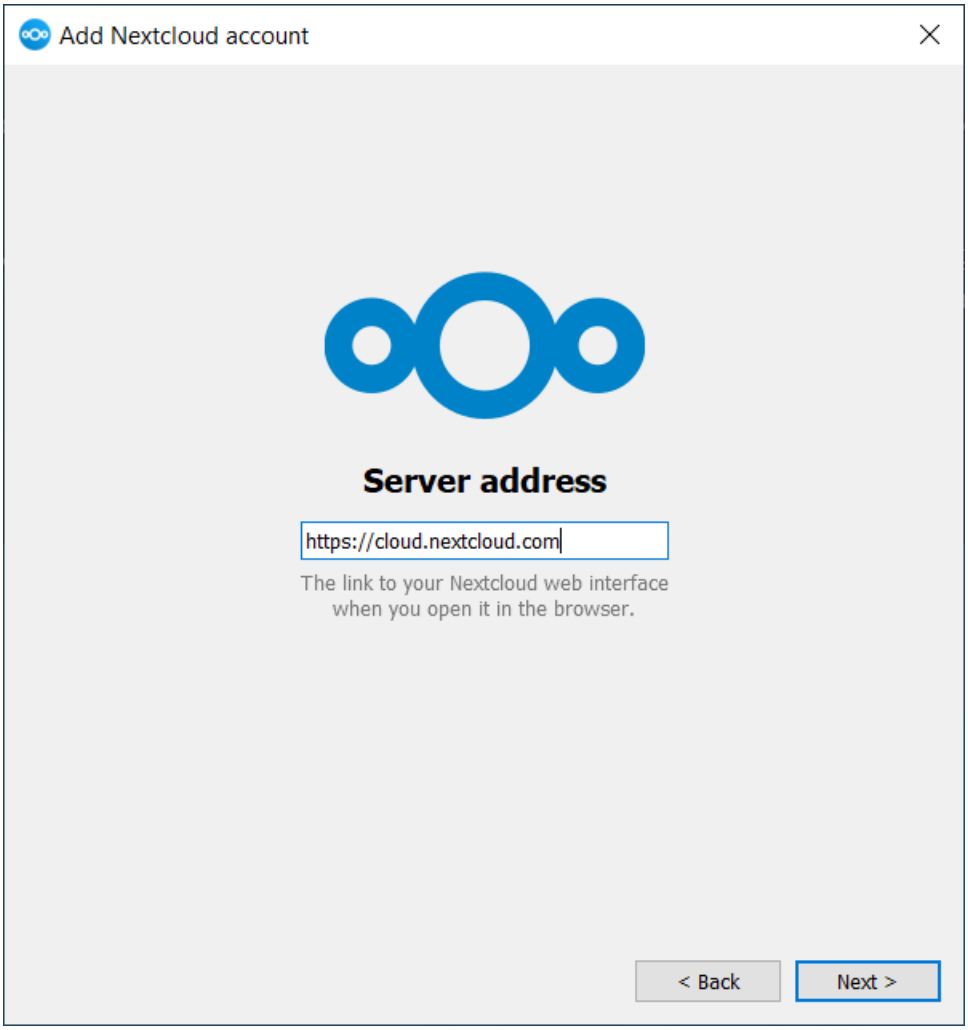
2.3 Installation Wizard

The installation wizard takes you step-by-step through configuration options and account setup. First, you need to enter the URL of your Nextcloud server.

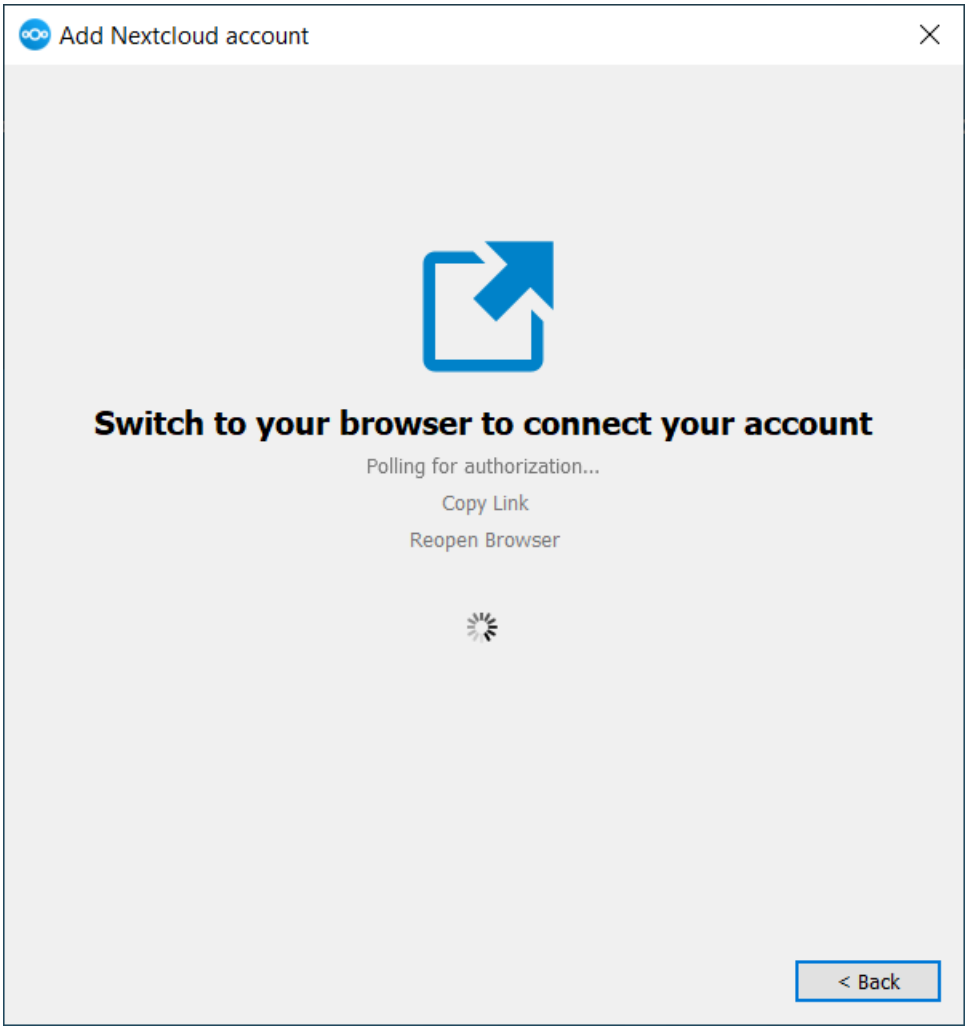


If you already have an account on a Nextcloud instance, you want to press the button `Login to your Nextcloud`. If you don't have a Nextcloud instance and an account there, you might want to register an account with a provider. Press

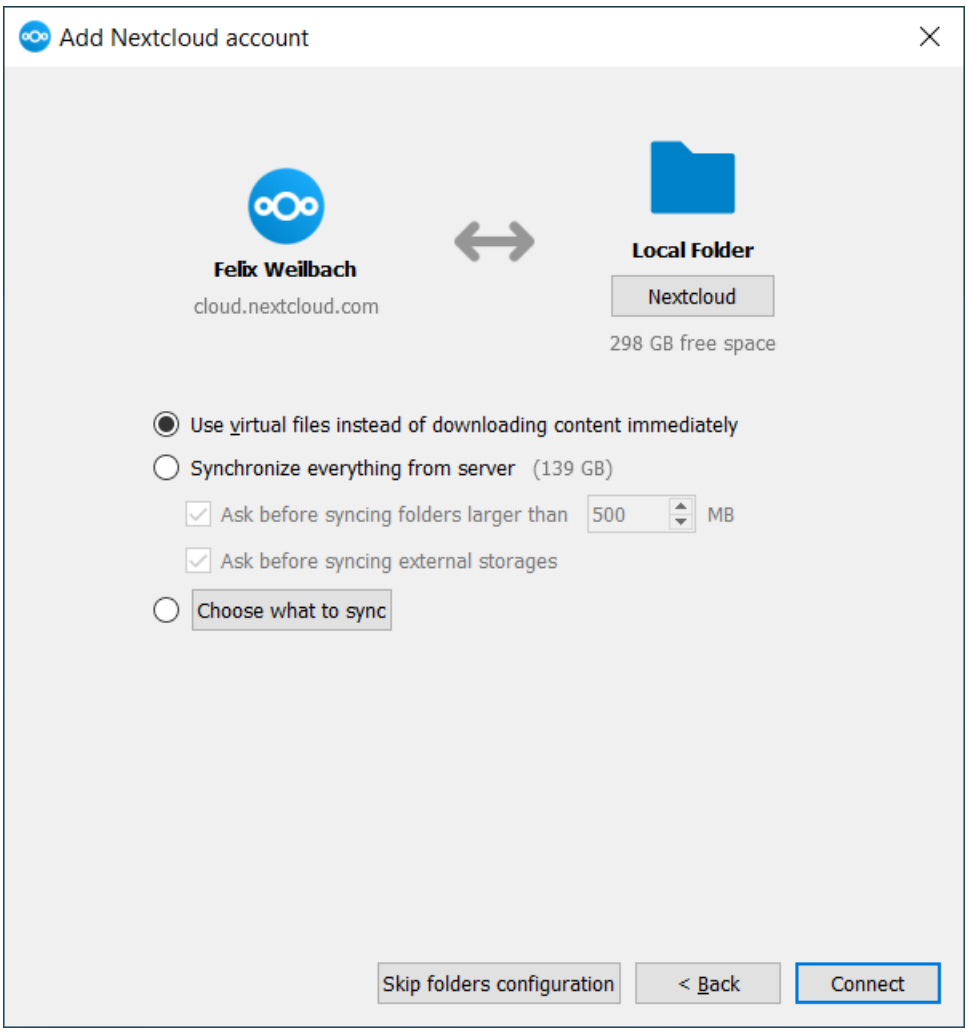
`Create account with Provider` in that case. Please keep in mind that the desktop client might have built without provider support. In that case, you won't see this page. Instead, you will be prompted with the next page.



Enter the URL for your Nextcloud instance. The URL is the same URL that you type into your browser when you try to access your Nextcloud instance.



Now your web browser should open and prompt you to login into your Nextcloud instance. Enter your username and password in your web browser and grant access. After you did that, go back to the wizard. Please keep in mind that you might not need to enter your username and password if you are already logged in to your browser.



On the local folder options screen, you may sync all of your files on the Nextcloud server, or select individual folders. The default local sync folder is `Nextcloud`, in your home directory. You may change this as well.

When you have completed selecting your sync folders, click the Connect button at the bottom right. The client will attempt to connect to your Nextcloud server, and when it is successful, the wizard closes itself. You can now observe the sync activity if you open the main dialogue by clicking on the tray icon.

CHAPTER

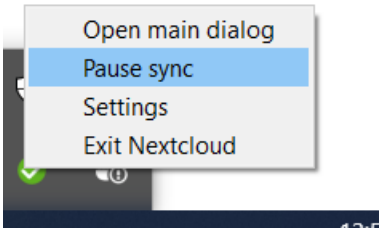
THREE

VISUAL TOUR

3.1 Icon

The Nextcloud Client remains in the background and is visible as an icon in the system tray (Windows, KDE), status bar (macOS), or notification area (Ubuntu).

3.2 Menu



A right click on the icon provides the following menu:

- `Open main dialog`: Opens the main dialog.
- `Pause sync`: Pauses the synchronization.
- `Settings`: Provides access to the settings menu.
- `Exit Nextcloud`: Quits Nextcloud Client, ending a currently running sync run.

Note: This menu is not available on macOS.

A double-click on the icon will open the currently-selected user's locally synced folder.

3.2.1 Settings

3.2.2 Account Settings

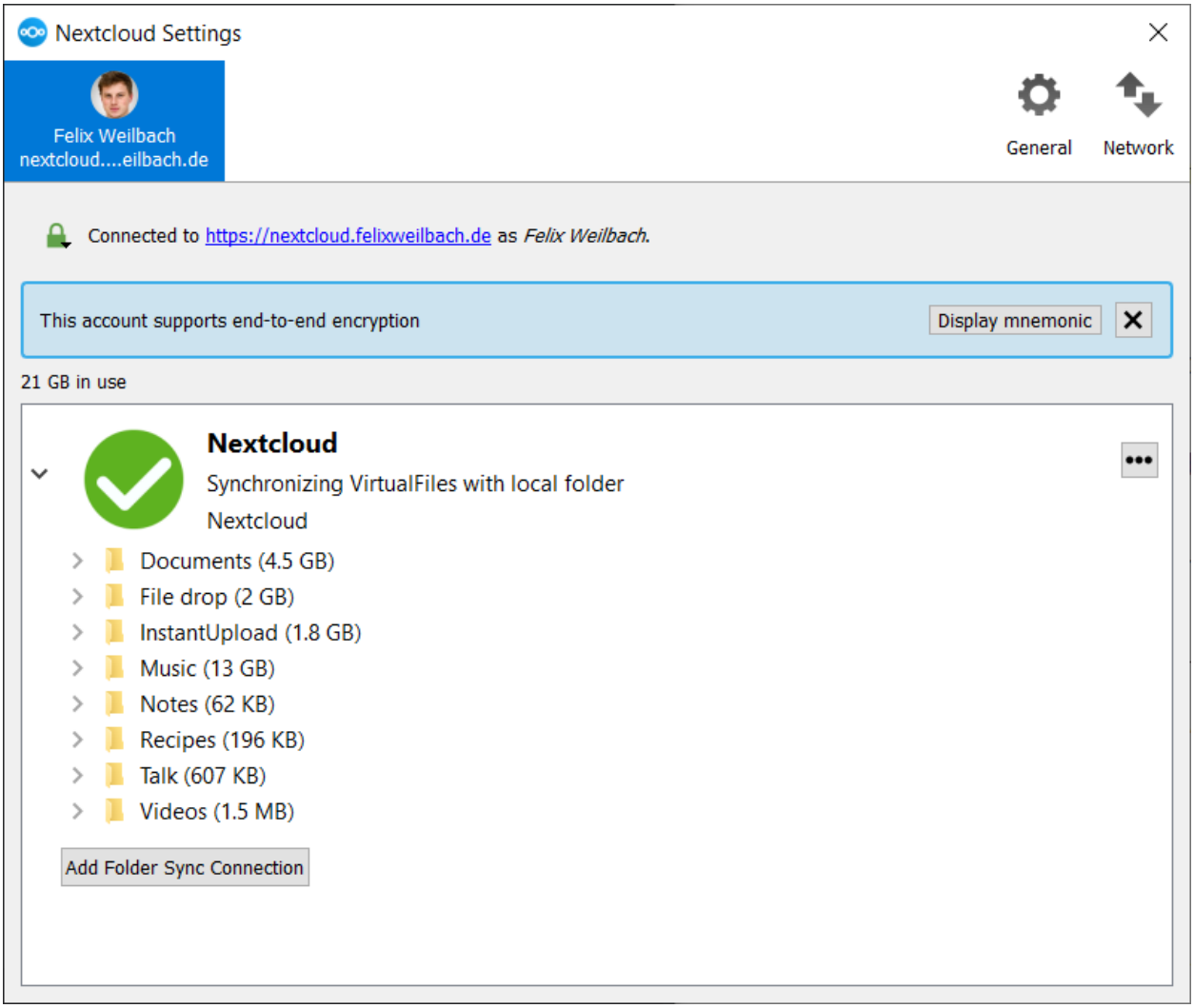
The `Account Settings tab` provides an executive summary about the synced folders in the account and gives the ability to modify them.

Where:

- `Connected to <Nextcloud instance> as <user>`: Indicates the Nextcloud server which the client is syncing with and the user account on that server.
- If the End-to-End encryption app is installed, then the current used passphrase can be shown with clicking on `Display mnemonic`.

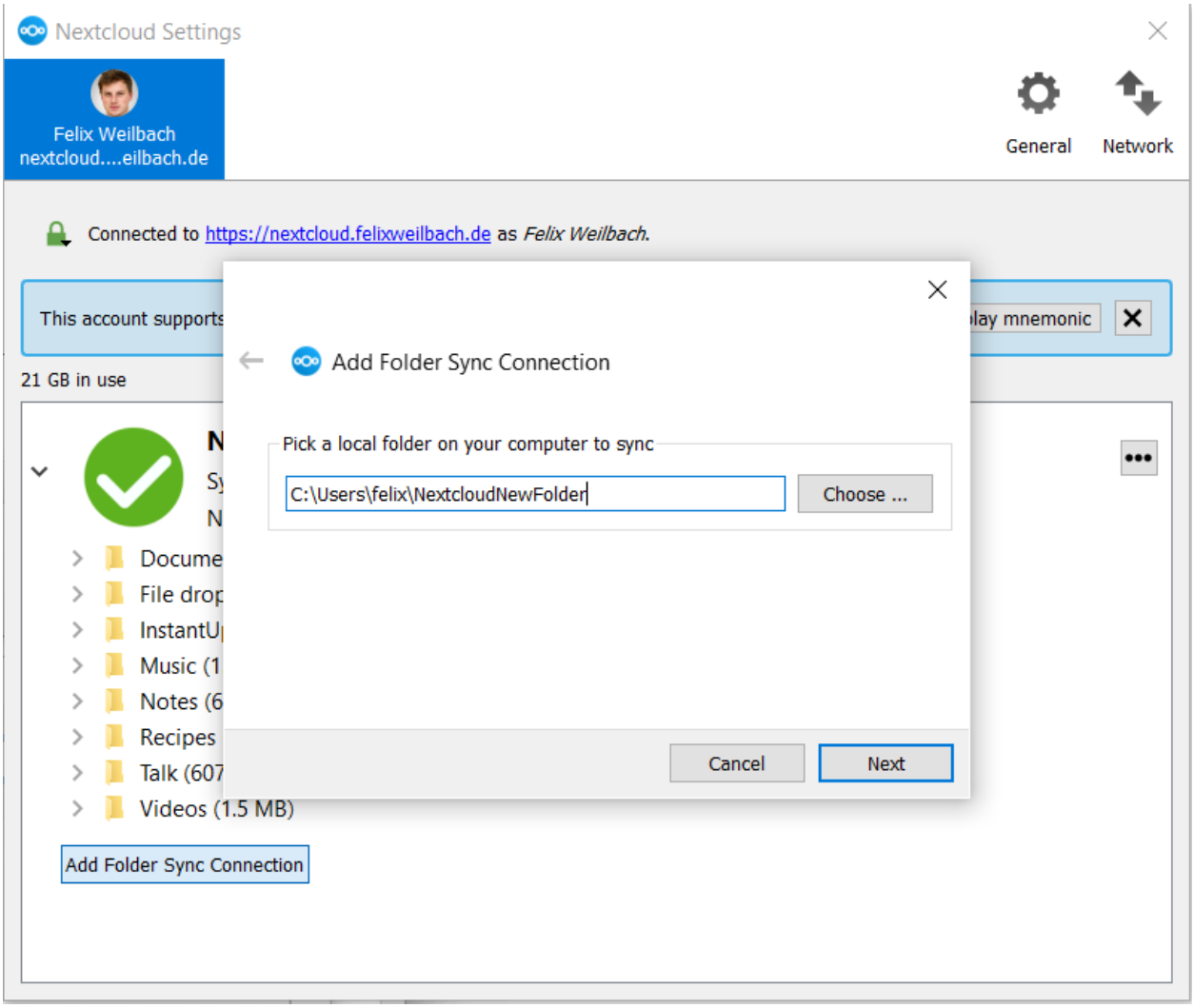
Clicking on the tree dot menu or doing a right click on the folder opens up a context menu with the following options:

- `Open folder`: Opens the folder in the file explorer which the operating system provides. In example on Windows it will open the Windows Explorer.
- `Create new folder`: Creates a new folder.
- `Force sync now`: Forces a synchronization run.
- `Availability`: This entry is only available if the folder uses the virtual files feature. The files in the folder can be either made available offline with `Make always available locally` or if disk space needs to be saved, they can be turned into placeholder files with `Free up local space`. `Free up local space` will not delete any files.
- `Add folder sync connection`: Provides the ability to add another folder to the sync (see `Adding a folder sync connection`).
- `Pause sync/Resume sync`: Will pause the current sync or prevent the client from starting a new sync. Resume will resume the sync process.
- `Remove folder sync connection`: Will remove the selected folder from being synced. This is used, for instance, when there is a desire to sync only a few folders and not the root. First, remove the root from sync, then add the folders to sync as desired.
- `Edit Ignored Files`: Provides a list of files which will be ignored, i.e., will not sync between the client and server. The ignored files editor allows adding patterns for files or directories that should be excluded from the sync process. Besides normal characters, wild cards may be used, an asterisk '*' indicating multiple characters, or a question mark '?' indicating a single character.
- `Enable virtual file support/Disable virtual file support`: Enable or disable virtual file support for this folder.



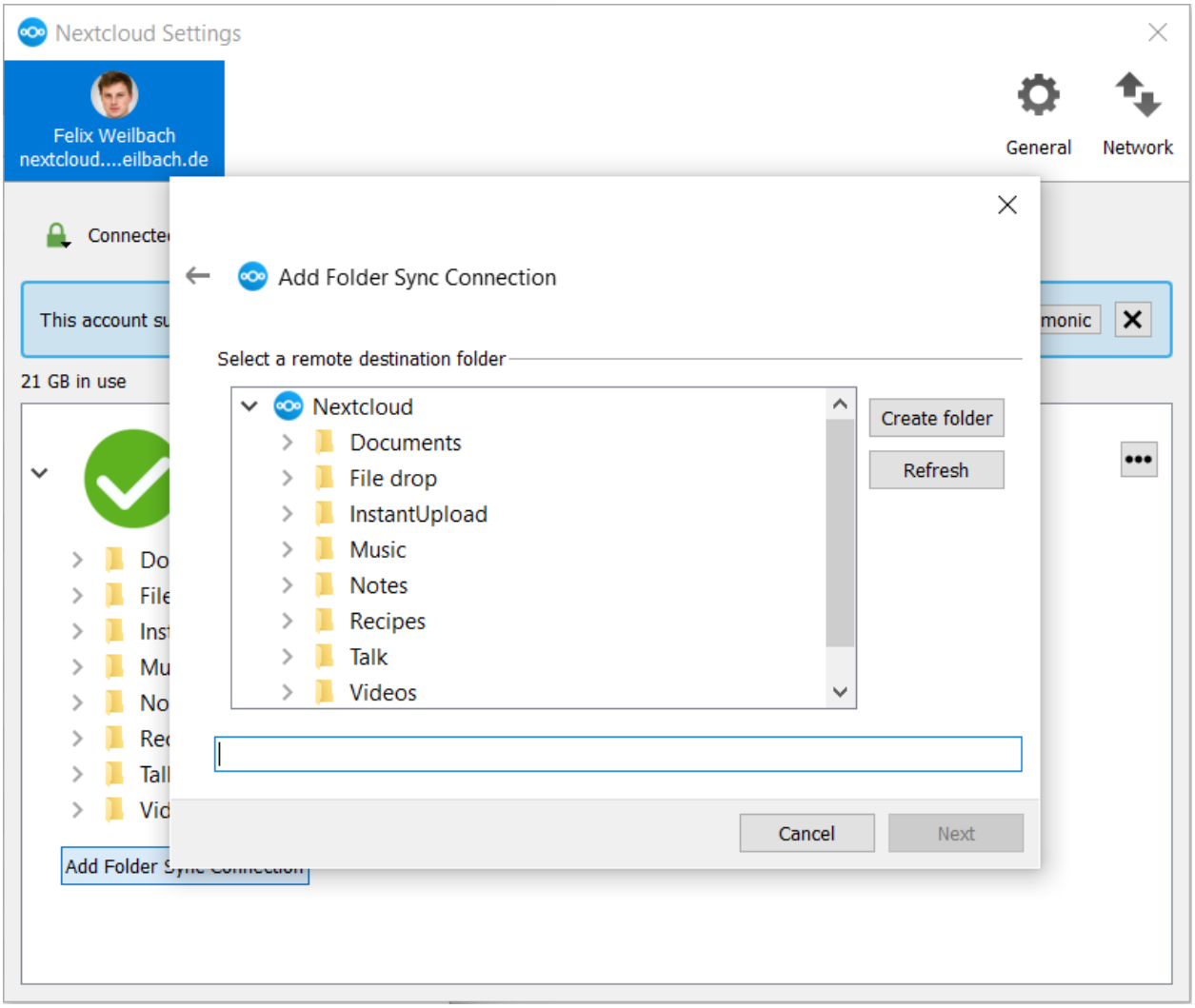
Adding a Folder Sync Connection

Adding a new sync is initiated by clicking `Add Folder Sync Connection` in the `Account` settings.



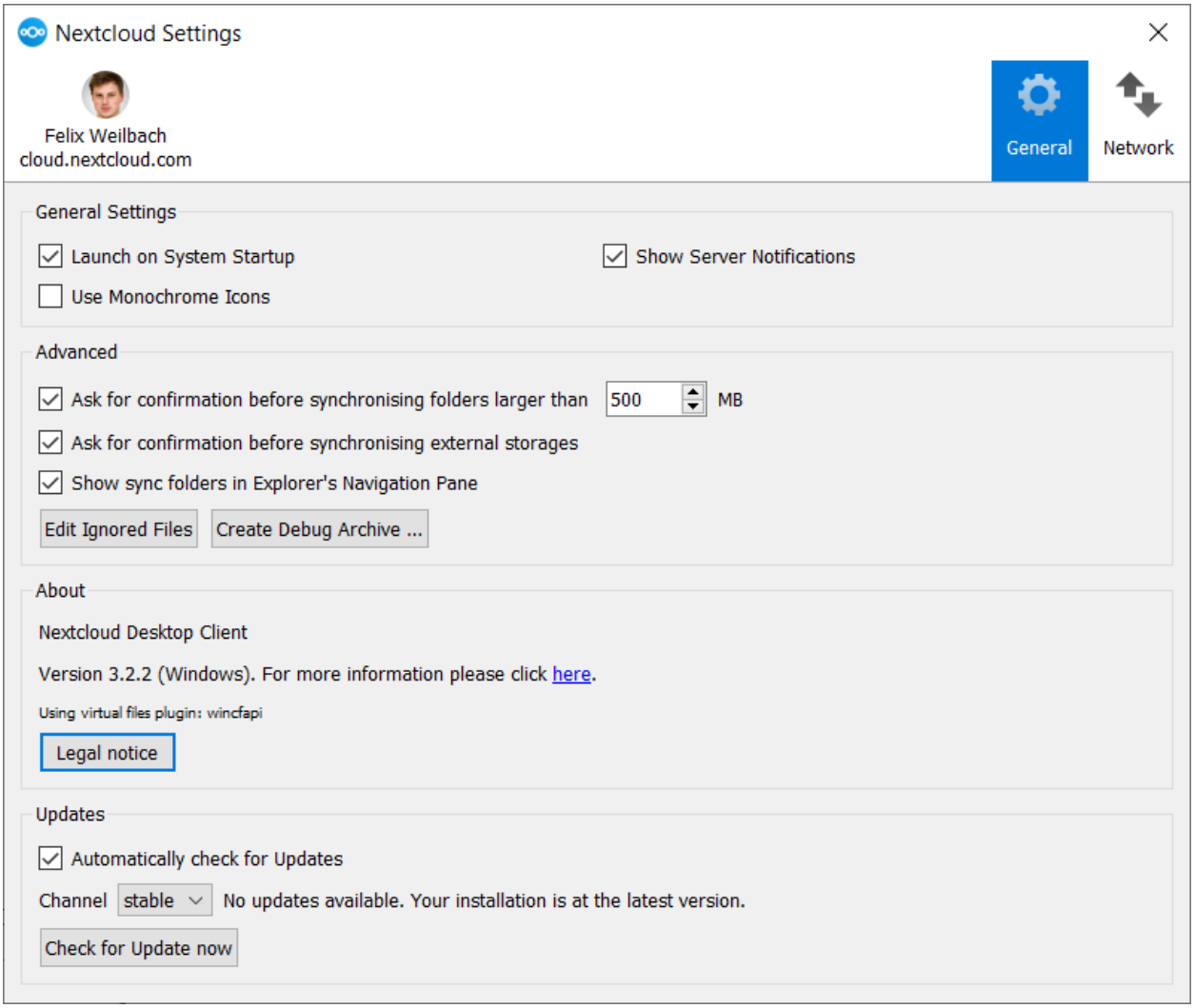
The Directory and alias name must be unique.

Then select the folder on the server to sync with. It is important to note that, a server folder can only sync to the client one time. So, in the above example, the sync is to the server root directory and thus it is not possible to select another folder under the root to sync.



3.2.3 General

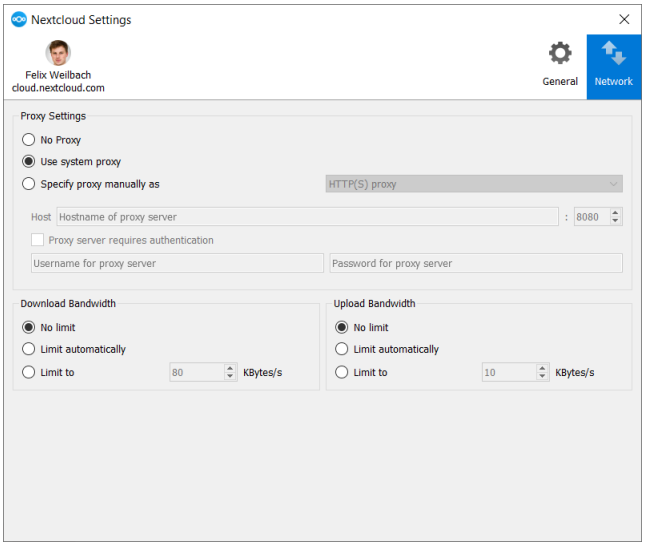
The tab provides several useful options:



- `Launch on System Startup`: This option is automatically activated once a user has conimaged his account. Un-checking the box will cause Nextcloud client to not launch on startup for a particular user.
- `Show Desktop Nofications`: When checked, bubble notifications when a set of sync operations has been performed are provided.
- `Use Monochrome Icons`: Use less obtrusive icons. Especially useful on macOS.
- `About`: provides information about authors as well as build conditions. This information is valuable when submitting a support request.

3.2.4 Network

This tab consolidates `Proxy Settings` and `Bandwidth Limiting`:



Proxy Settings

- `No Proxy`: Check this if Nextcloud Client should circumvent the default proxy conimaged on the system.
- `Use system proxy`: Default, will follow the systems proxy settings. On Linux, this will only pick up the value of the variable `http_proxy`.
- `Specify proxy manually as`: Allows to specify custom proxy settings. If you require to go through a HTTP(S) proxy server such as Squid or Microsoft Forefront TMG, pick `HTTP(S)`. `SOCKSv5` on the other hand is particularly useful in special company LAN setups, or in combination with the OpenSSH dynamic application level forwarding feature (see `ssh -D`).
- `Host`: Enter the host name or IP address of your proxy server, followed by the port number. HTTP proxies usually listen on Ports 8080 (default) or 3128. SOCKS server usually listen on port 1080.
- `Proxy Server requires authentication`: Should be checked if the proxy server does not allow anonymous usage. If checked, a username and password must be provided.

Bandwidth Limiting

The Download Bandwidth can be either unlimited (default) or limited to a custom value. This is the bandwidth available for data flowing from the Nextcloud Server to the client.

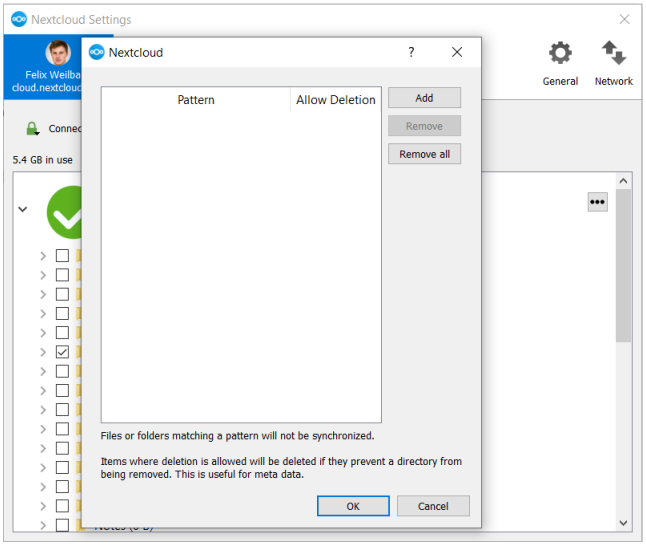
The Upload Bandwidth, the bandwidth available or data flowing from the Nextcloud client to the server, has an additional option to limit automatically.

When this option is checked, the Nextcloud client will surrender available bandwidth to other applications. Use this option if there are issues with real time communication in conjunction with the Nextcloud Client.

3.2.5 The Ignored Files Editor

Nextcloud Client has the ability to exclude files from the sync process. The ignored files editor allows editing of custom patterns for files or directories that should be excluded from the sync process.

There is a system wide list of default ignore patterns. These global defaults cannot be directly modified within the editor. Hovering with the mouse will reveal the location of the global exclude definition file.



Each line contains an ignore pattern string. Next to normal characters, wildcards can be used to match an arbitrary number of characters, designated by an asterisk (*) or a single character, designated by a question mark (?). If a pattern ends with a slash character (/) the pattern is only applied to directory components of the path to check.

If the checkbox is checked for a pattern in the editor it means that files which are matched by this pattern are fleeting metadata which the client will *remove*.

Note: Modifying the global exclude definition file might render the client unusable or cause undesired behavior.

Note: Custom entries are currently not validated for syntactical correctness by the editor, but might fail to load correctly.

In addition to this list, Nextcloud Client always excludes files with characters that cannot be synced to other file systems.

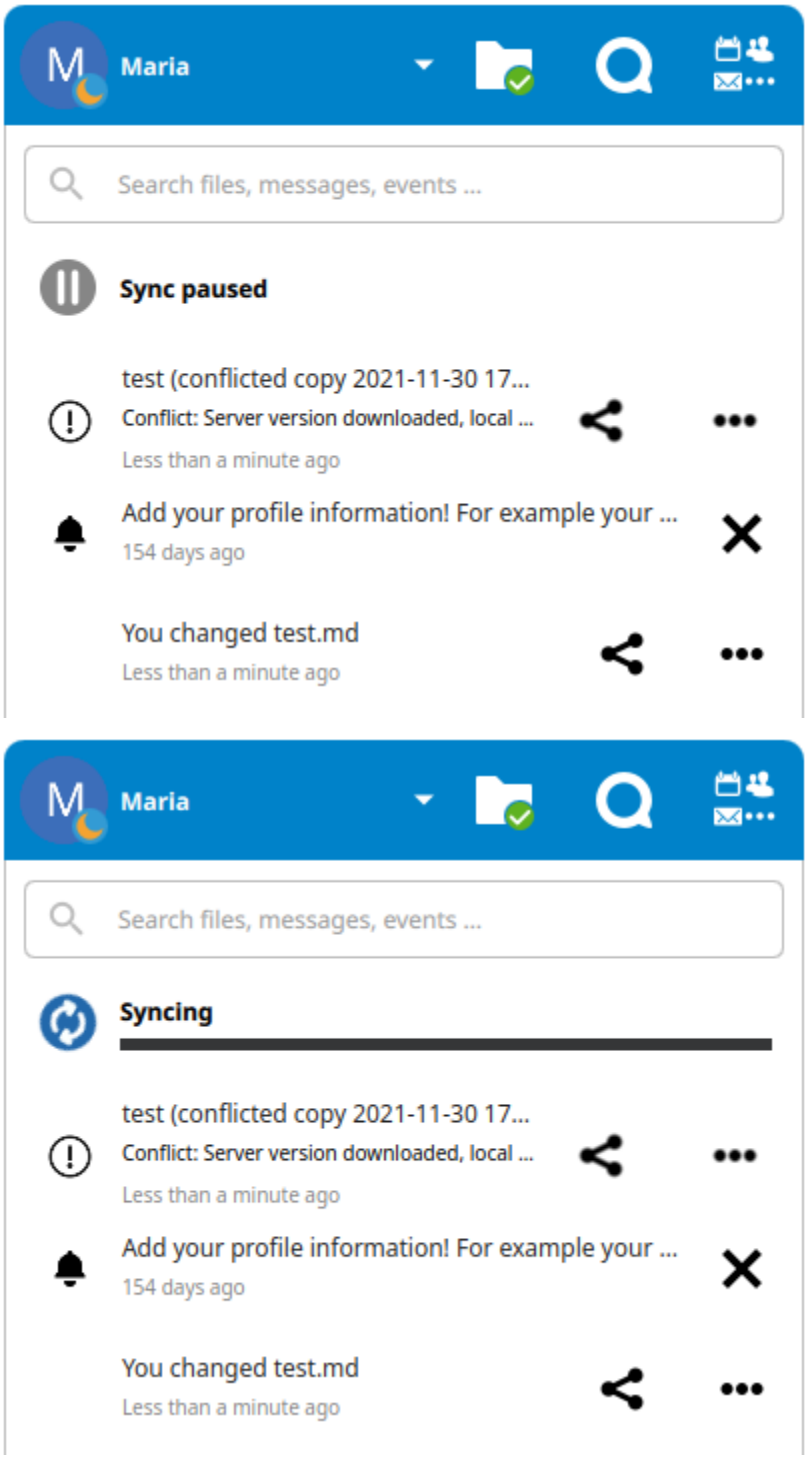
With version 1.5.0 it also ignores files that caused individual errors while syncing for a three times. These are listed in the activity view. There also is a button to retry the sync for another three times.

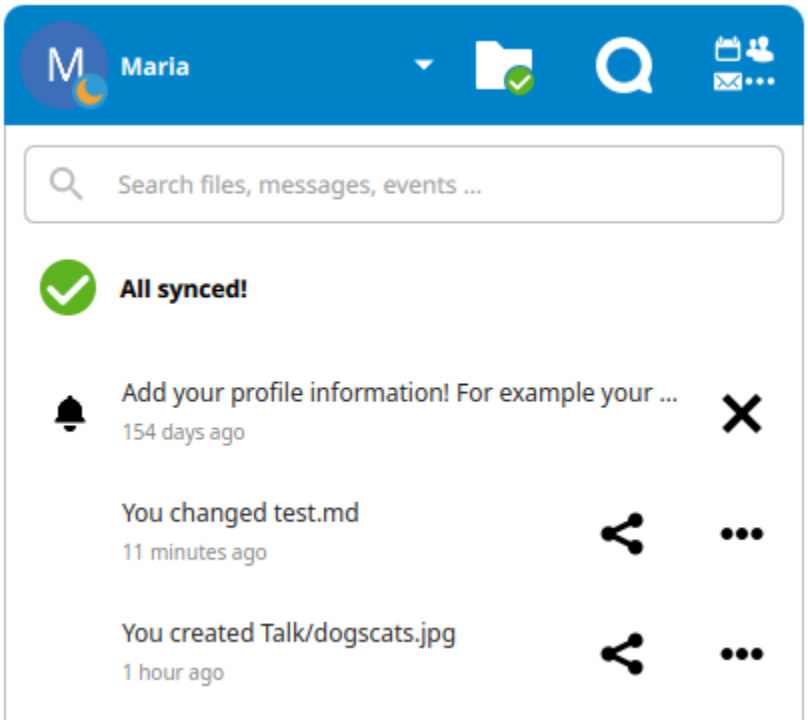
For more detailed information see *Ignored Files*.

3.3 Main dialog

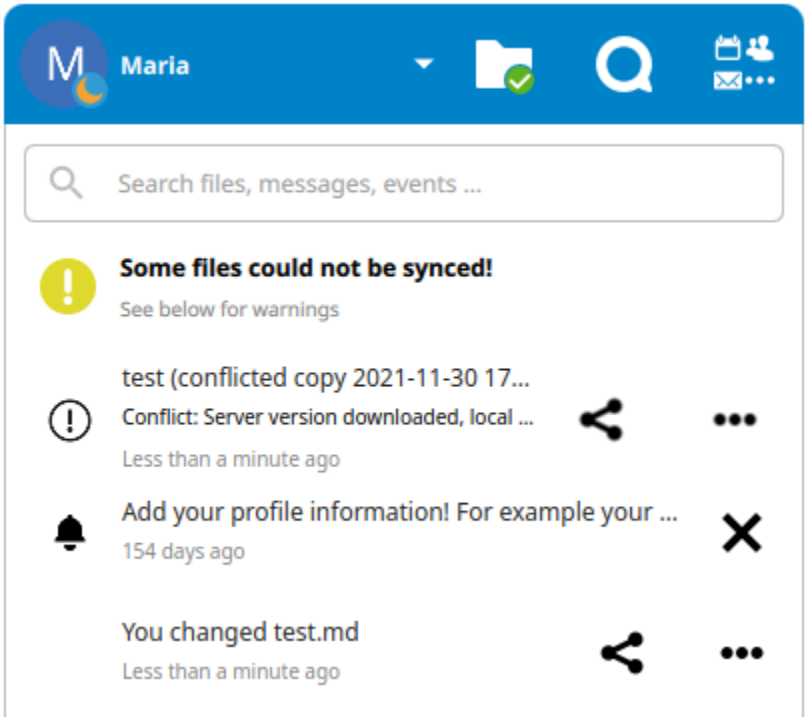
3.3.1 Sync State

The main dialog, which can be invoked from the tray icon in the taskbar, will show files information about the activities of the sync client and Nextcloud.



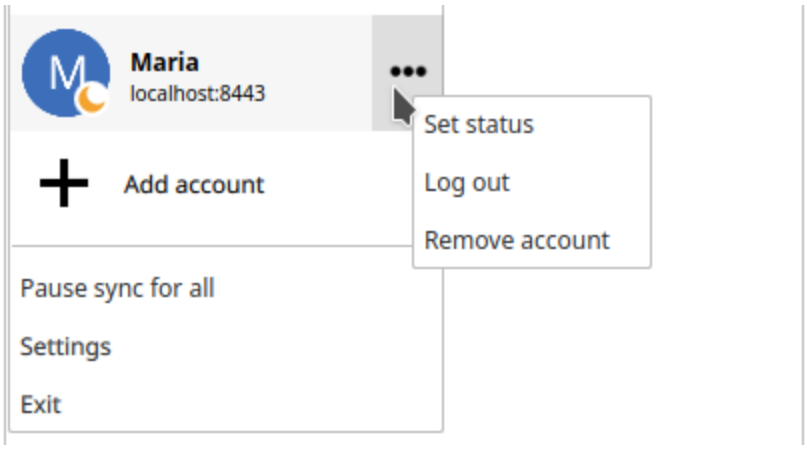


If there are any synchronization issues, they will show up here:



For more information on how to solve these issues see *Appendix C: Troubleshooting*.

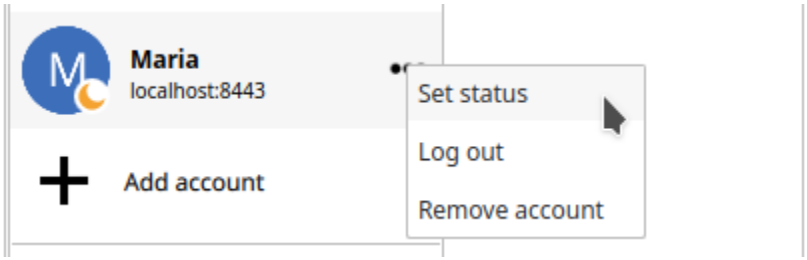
When clicking on the avatar a menu opens where it is possible to add a new account or removing an existing account.

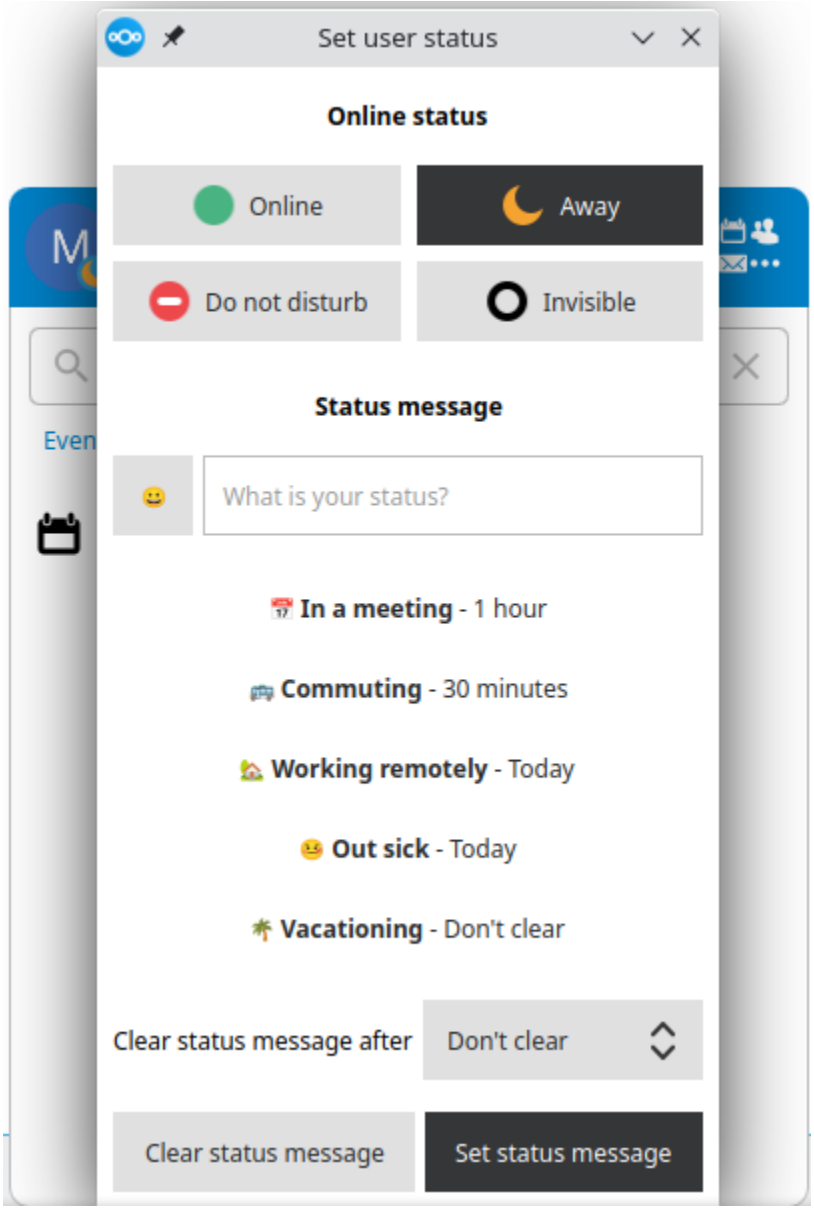


3.3.2 User Status

User status is displayed in the Nextcloud desktop client's tray window. Default user status is always "Online" if no other status is available from the server-side.

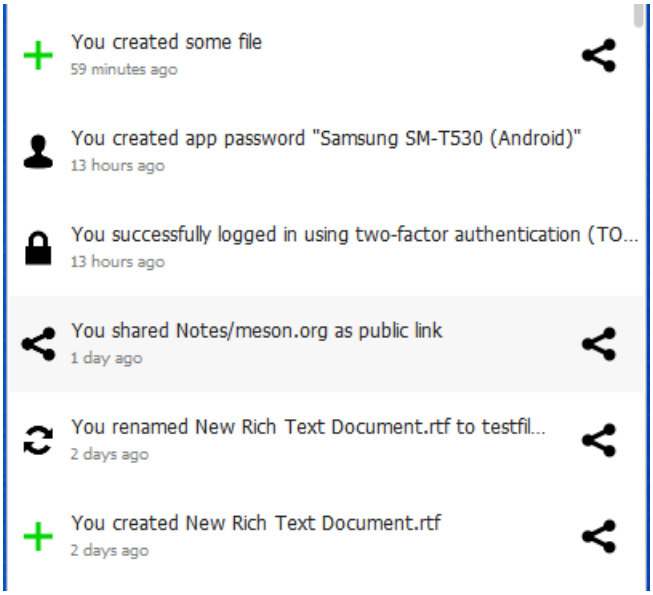
When clicking on `Set status` you can edit the emoji, message and the timer to clear your user status:





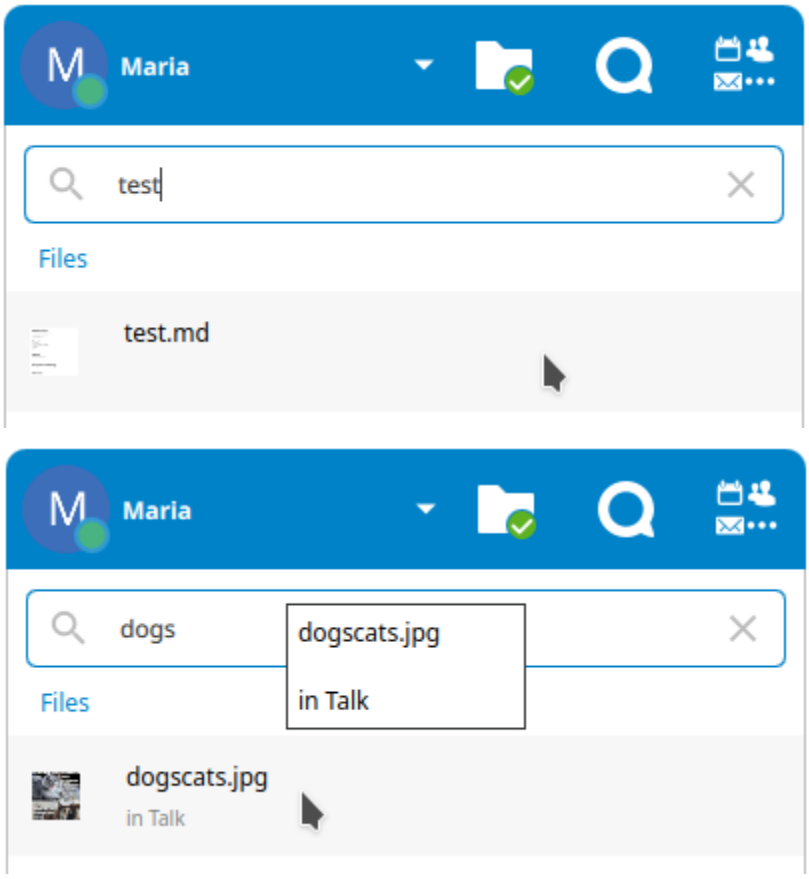
3.3.3 Activities list

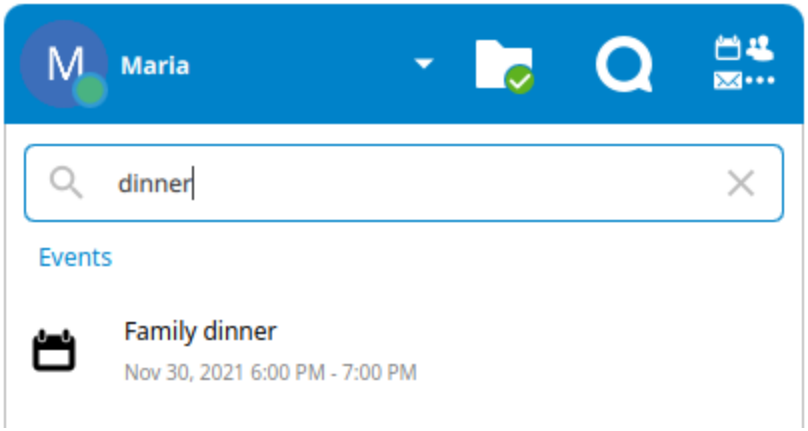
The dialog also gives information about other activities or notifications like Talk mentions or file changes. It does also show the status of the user.



3.3.4 Unified search

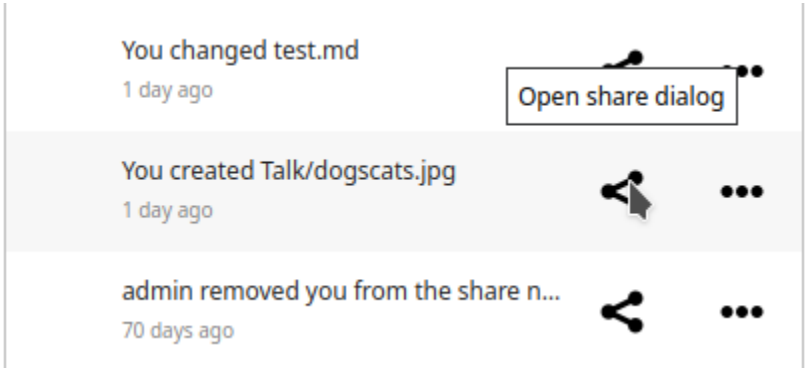
With the unified search you can find everything you have in your server - files, Talk messages, calendar appointments:

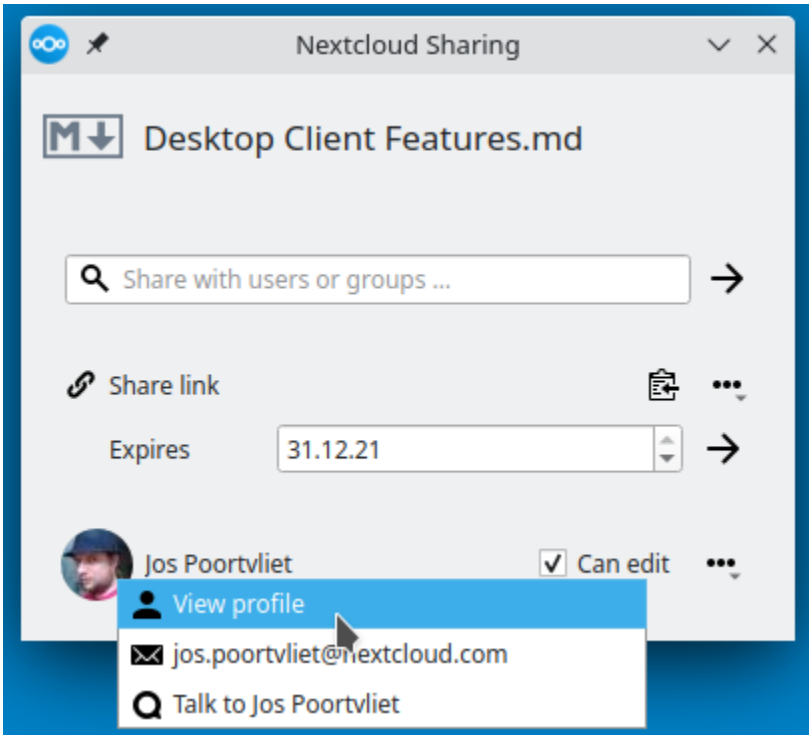




3.3.5 Share dialog: Talk options and View Profile

You can now share a file directly in a conversation in Talk and view the sharee user profile:




Nextcloud Sharing
Desktop Client Features.md
Share with users or groups ...
Share link
Expires
31.12.21
Jos Poortvliet
Can edit
View profile
jos.poortvliet@nextcloud.com
Talk to Jos Poortvliet

CHAPTER

FOUR

USING THE SYNCHRONIZATION CLIENT

The Nextcloud Desktop Client remains in the background and is visible as an icon in the system tray (Windows, KDE), menu bar (Mac OS X), or notification area (Linux).

The status indicator uses icons to indicate the current status of your synchronization. The green circle with the white checkmark tells you that your synchronization is current and you are connected to your Nextcloud server.

The blue icon with the white semi-circles means synchronization is in progress.

The yellow icon with the parallel lines tells you your synchronization has been paused. (Most likely by you.)

The gray icon with three white dots means your sync client has lost its connection with your Nextcloud server.

When you see a yellow circle with the sign "!" that is the informational icon, so you should click it to see what it has to tell you.

The red circle with the white "x" indicates a configuration error, such as an incorrect login or server URL.

4.1 Systray Icon

A right-click on the systray icon opens a menu for quick access to multiple operations.

This menu provides the following options:

- Open main dialog
- Paus sync/Resume sync
- Settings

- Exit Nextcloud, logging out and closing the client

A left-click on your systray icon opens the main dialog of the desktop client.

The main dialogs show recent activities, errors and server notifications.

When clicking on the main dialog and then clicking on the avatar of the user, the Settings can be opened.

4.1.1 Configuring Nextcloud Account Settings

At the top of the window are tabs for each configured sync account, and two others for General and Network settings. On your account tabs you have the following features:

- Connection status, showing which Nextcloud server you are connected to, and your Nextcloud username.
- Used and available space on the server.
- Current synchronization status.
- **Add Folder Sync Connection** button.

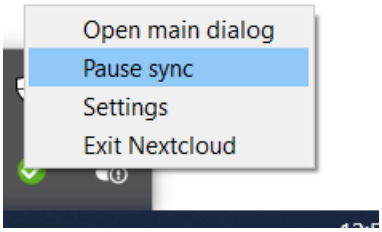
The little button with three dots (the overflow menu) that sits to the right of the sync status bar offers additional options:

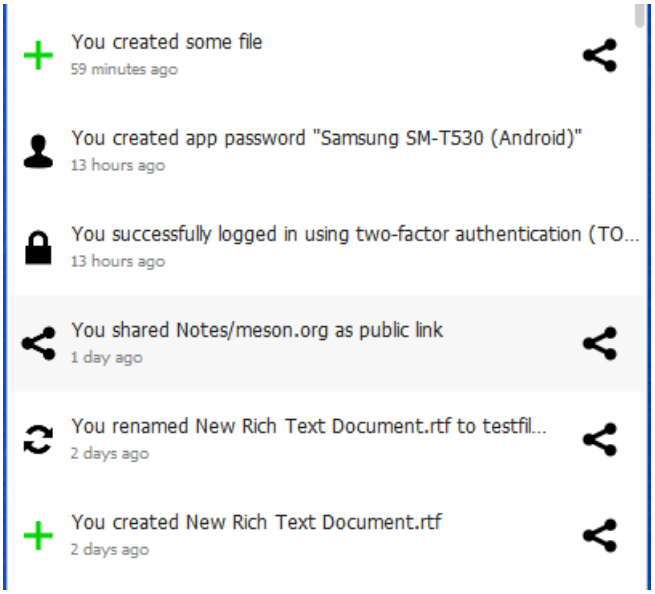
- Open Folder
- Choose What to Sync (This appears only when your file tree is collapsed, and expands the file tree)
- Pause Sync / Resume Sync
- Remove folder sync connection
- Availability (Only available if virtual files support is enabled)
- Enable virtual file support/Disable virtual file support

Open Folder opens your local Nextcloud sync folder.

Pause Sync pauses sync operations without making any changes to your account. It will continue to update file and folder lists, without downloading or updating files. To stop all sync activity use **Remove Folder Sync Connection**.

Note: Nextcloud does not preserve the mtime (modification time) of directories, though it does update the mtimes on files. See Wrong folder date when syncing for discussion of this.





4.1.2 Adding New Accounts

You may configure multiple Nextcloud accounts in your desktop sync client. Simply click the **Account** > **Add New** button on any account tab to add a new account, and then follow the account creation wizard. The new account will appear as a new tab in the settings dialog, where you can adjust its settings at any time. Use **Account** > **Remove** to delete accounts.

4.2 File Manager Overlay Icons

The Nextcloud sync client provides overlay icons, in addition to the normal file type icons, for your system file manager (Explorer on Windows, Finder on Mac and Nautilus on Linux) to indicate the sync status of your Nextcloud files.

The overlay icons are similar to the systray icons introduced above. They behave differently on files and directories according to sync status and errors.

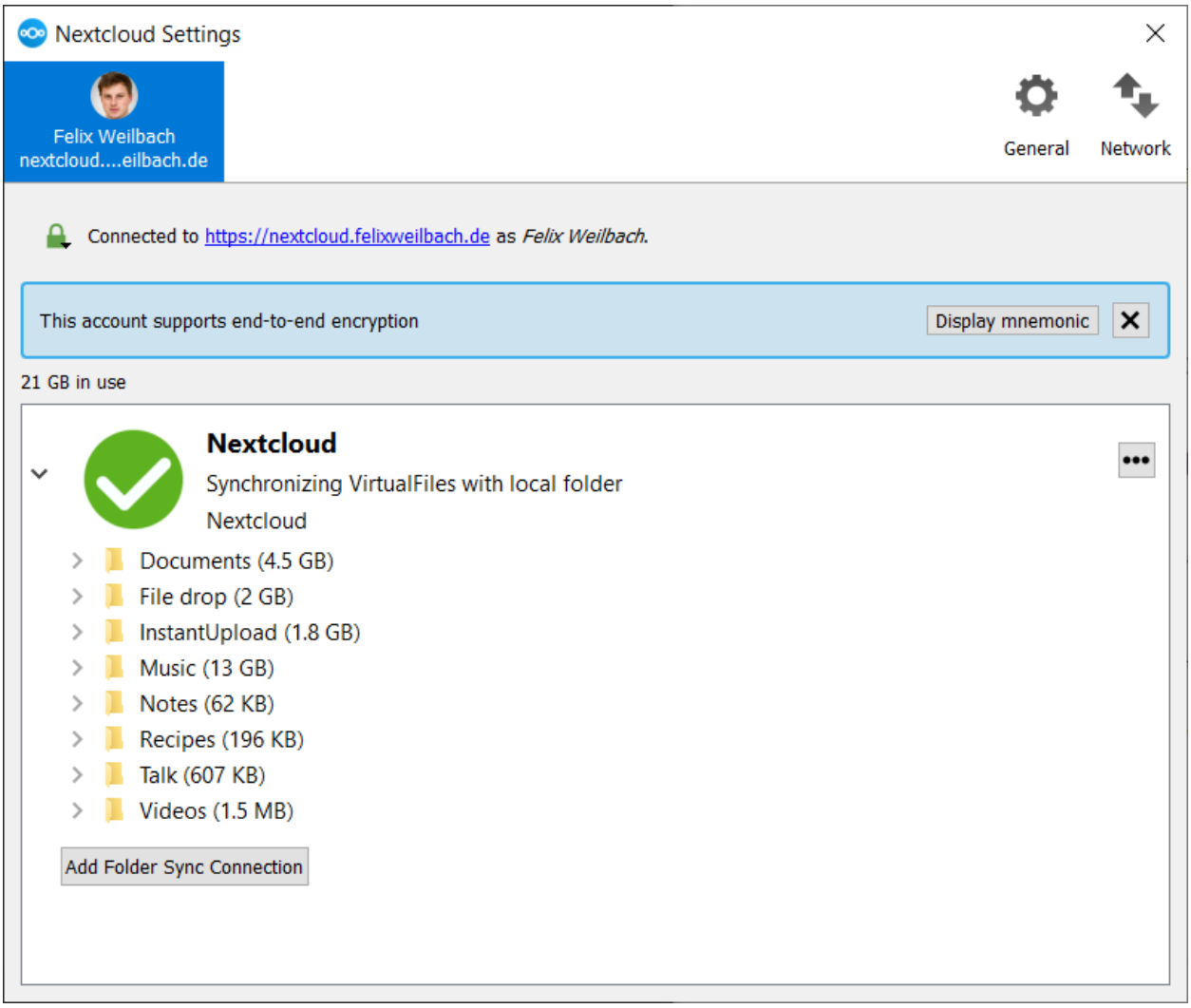
The overlay icon of an individual file indicates its current sync state. If the file is in sync with the server version, it displays a green checkmark.

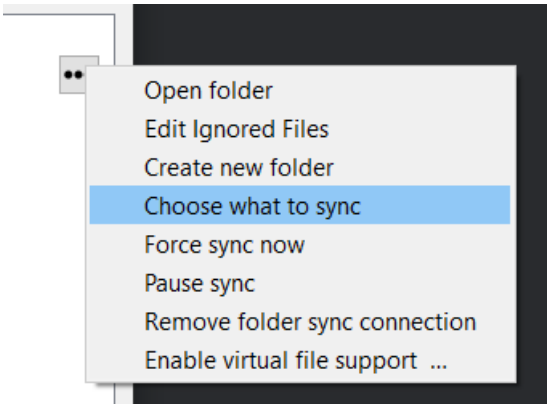
If the file is ignored from syncing, for example because it is on your exclude list, or because it is a symbolic link, it displays a warning icon.

If there is a sync error, or the file is blacklisted, it displays an eye-catching red X.

If the file is waiting to be synced, or is currently syncing, the overlay icon displays a blue cycling icon.

When the client is offline, no icons are shown to reflect that the folder is currently out of sync and no changes are synced to the server.


Nextcloud Settings
Felix Weilbach
nextcloud....eilbach.de
General
Network
Connected to https://nextcloud.felixweilbach.de as Felix Weilbach.
This account supports end-to-end encryption
Display mnemonic
21 GB in use
Nextcloud
Synchronizing VirtualFiles with local folder
Nextcloud
Documents (4.5 GB)
File drop (2 GB)
InstantUpload (1.8 GB)
Music (13 GB)
Notes (62 KB)
Recipes (196 KB)
Talk (607 KB)
Videos (1.5 MB)
Add Folder Sync Connection

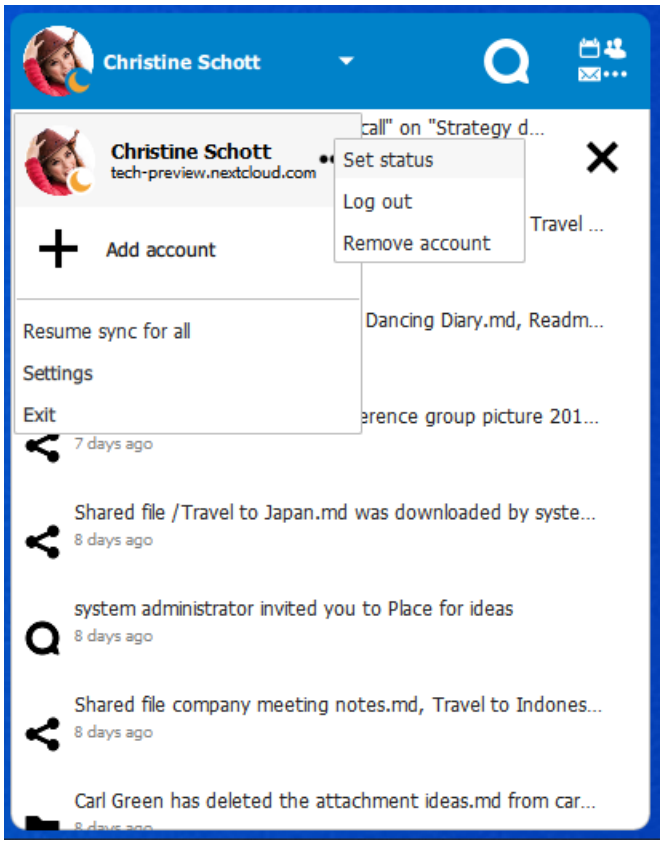

Open folder
Edit Ignored Files
Create new folder
Choose what to sync
Force sync now
Pause sync
Remove folder sync connection
Enable virtual file support ...

The overlay icon of a synced directory indicates the status of the files in the directory. If there are any sync errors, the directory is marked with a warning icon.

If a directory includes ignored files that are marked with warning icons that does not change the status of the parent directories.

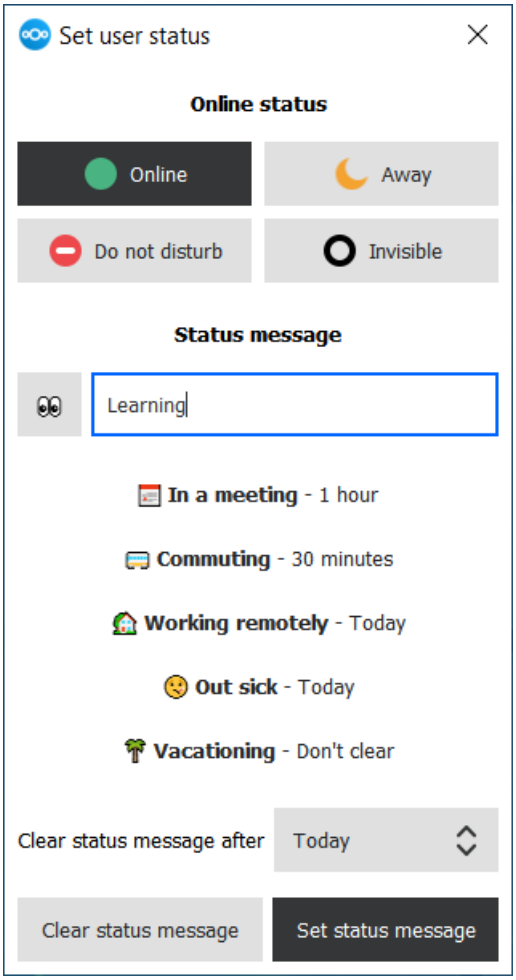
4.3 Set the user status

If you have the user status app installed on your Nextcloud server, you can set your user status from the desktop client. To do so, open the main dialog. Then click on your avatar and then click on the three dots. In the menu that opens click on **Set status**.



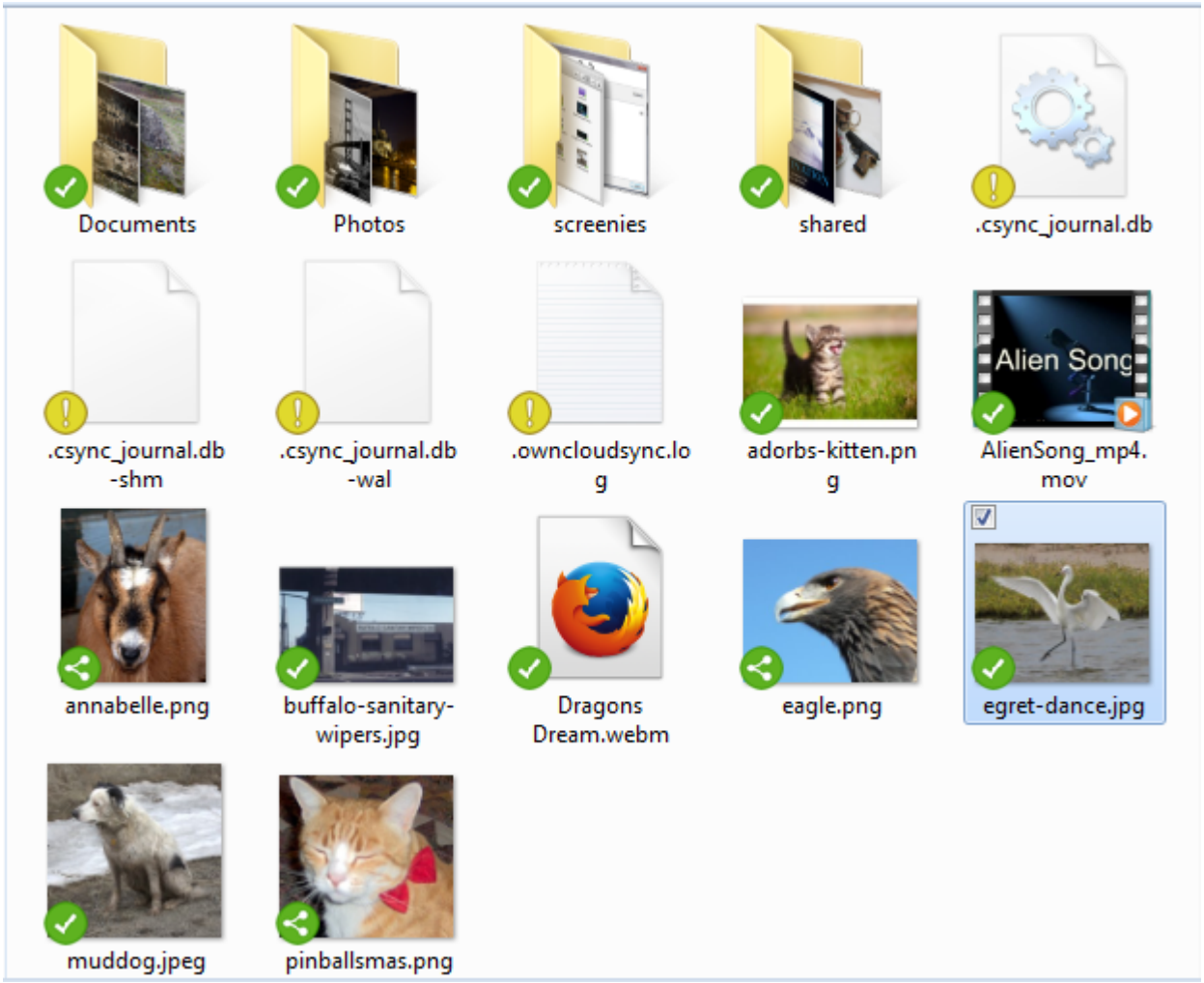
In the dialog that opens, you can set your online status if you click on either **Online**, **Away**, **Do not disturb** or **Invisible**. You can also set a custom status message with the text field below or choose one of the predefined status messages below. It is also possible to set a custom emoji if you click on the button with the emoji beside the text input field. The last thing you might want to set is when your user status should be cleared. You can choose the period after which the user status will be cleared by clicking on the button on the left hand side of the text **Clear status message after**.

If you are happy with the status you have created you can enable this status with the button **Set status message**. If you had already a status set, you can clear the status by clicking the cutton **Clear status message**.


Set user status
Online status
Online
Away
Do not disturb
Invisible
Status message
Learning
In a meeting - 1 hour
Commuting - 30 minutes
Working remotely - Today
Out sick - Today
Vacationing - Don't clear
Clear status message after
Today
Clear status message
Set status message

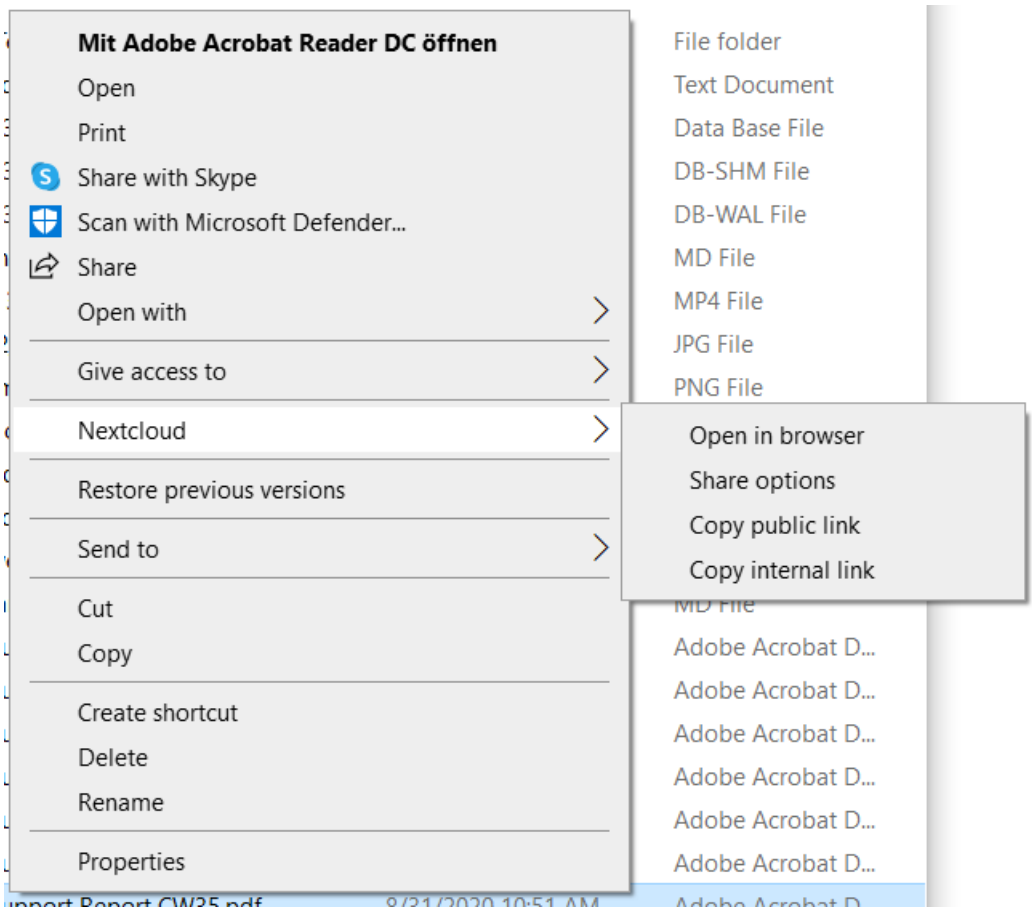
4.4 Sharing From Your Desktop

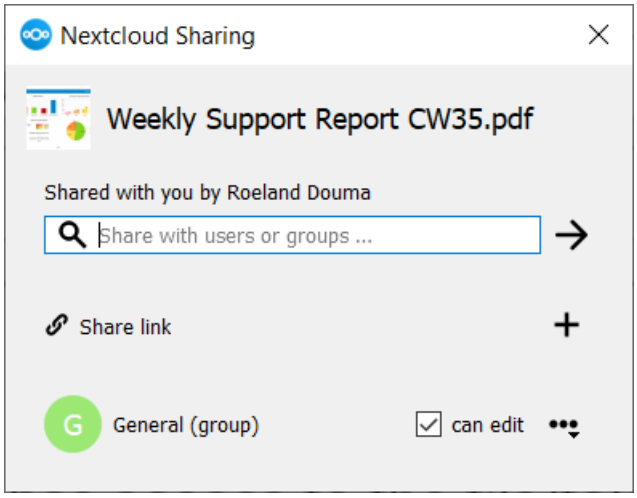
The Nextcloud desktop sync client integrates with your file manager. Finder on macOS and Explorer on Windows. Linux users must install an additional package depending on the used file manager. Available are e.g. `nautilus-nextcloud` (Ubuntu/Debian), `dolphin-nextcloud` (Kubuntu), `nemo-nextcloud` and `caja-nextcloud`. You can create share links, and share with internal Nextcloud users the same way as in your Nextcloud Web interface.



In you file explorer, click on a file and in the context menu go to **Nextcloud** and then lick on **Share options** to bring up the Share dialog.

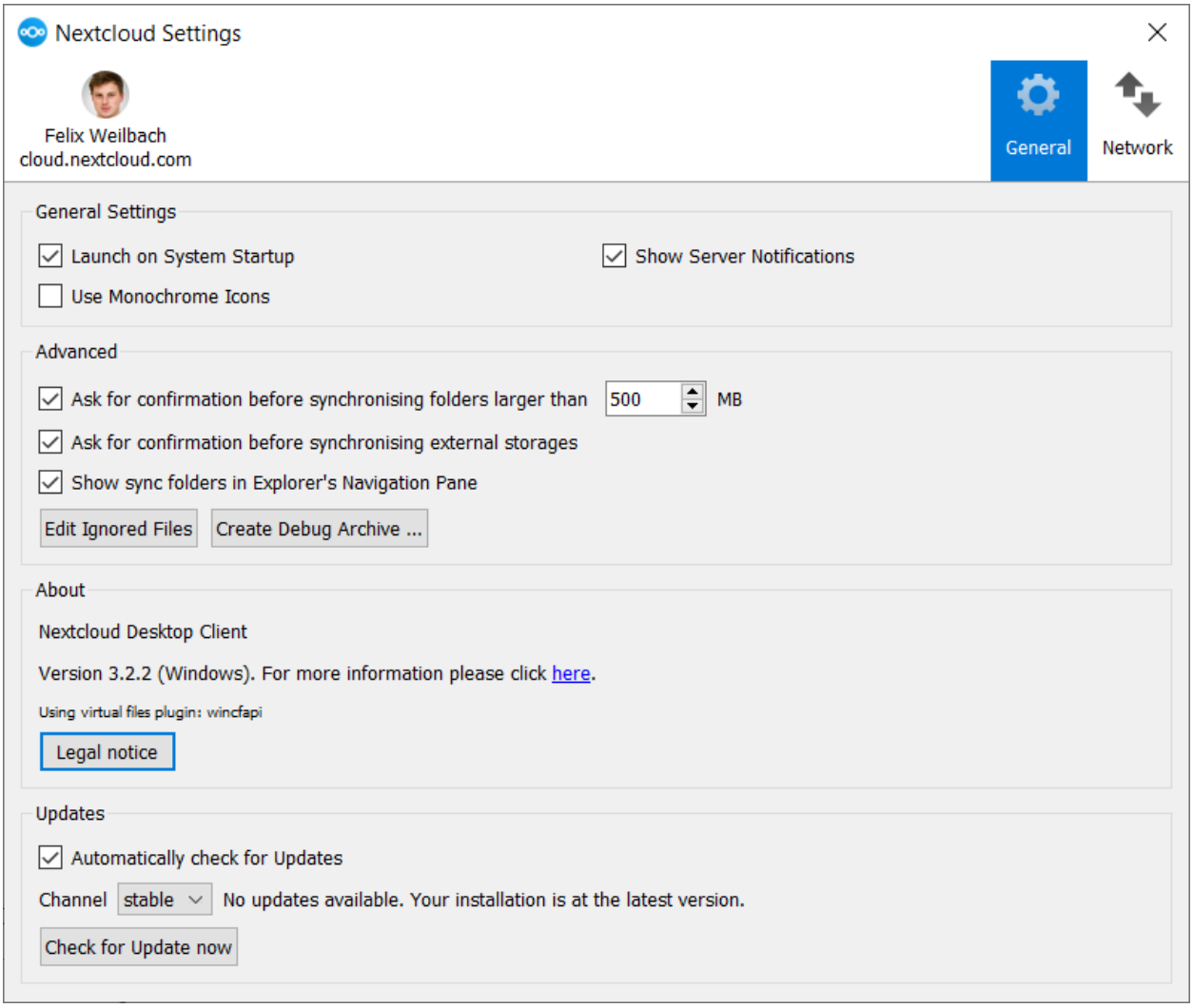
From this dialog you can share a file.


Mit Adobe Acrobat Reader DC öffnen
Open
Print
Share with Skype
Scan with Microsoft Defender...
Share
Open with
Give access to
Nextcloud
Restore previous versions
Send to
Cut
Copy
Create shortcut
Delete
Rename
Properties
Open in browser
Share options
Copy public link
Copy internal link
File folder
Text Document
Data Base File
DB-SHM File
DB-WAL File
MD File
MP4 File
JPG File
PNG File
Adobe Acrobat D...
Adobe Acrobat D...
Adobe Acrobat D...
Adobe Acrobat D...
Adobe Acrobat D...
Adobe Acrobat D...


Nextcloud Sharing
Weekly Support Report CW35.pdf
Shared with you by Roeland Douma
Share with users or groups ...
Share link
G
General (group)
can edit

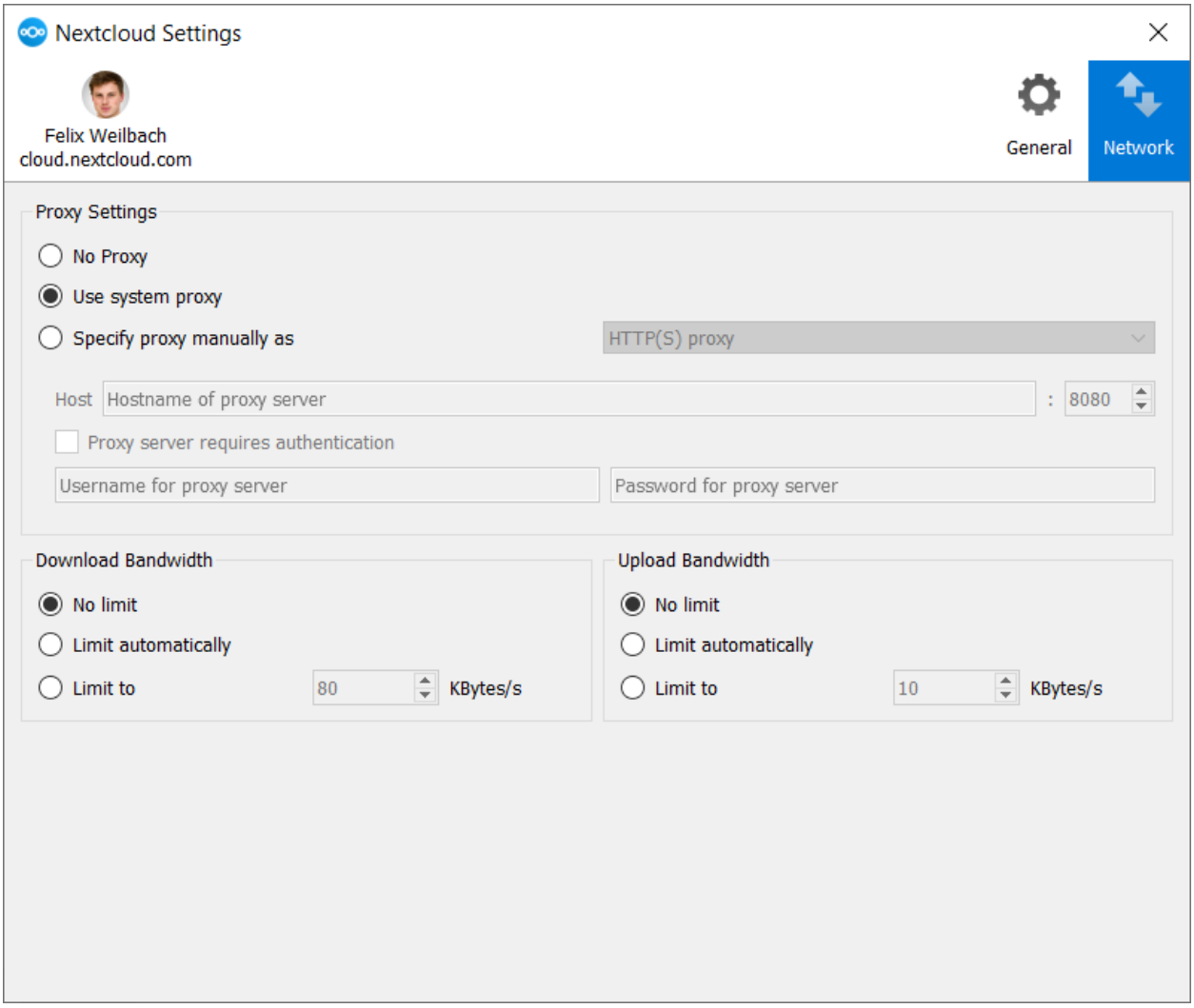
4.5 General Window

The General window has configuration options such as **Launch on System Startup**, **Use Monochrome Icons**, and **Show Desktop Notifications**. This is where you will find the **Edit Ignored Files** button, to launch the ignored files editor, and **Ask confirmation before downloading folders larger than [folder size]**.



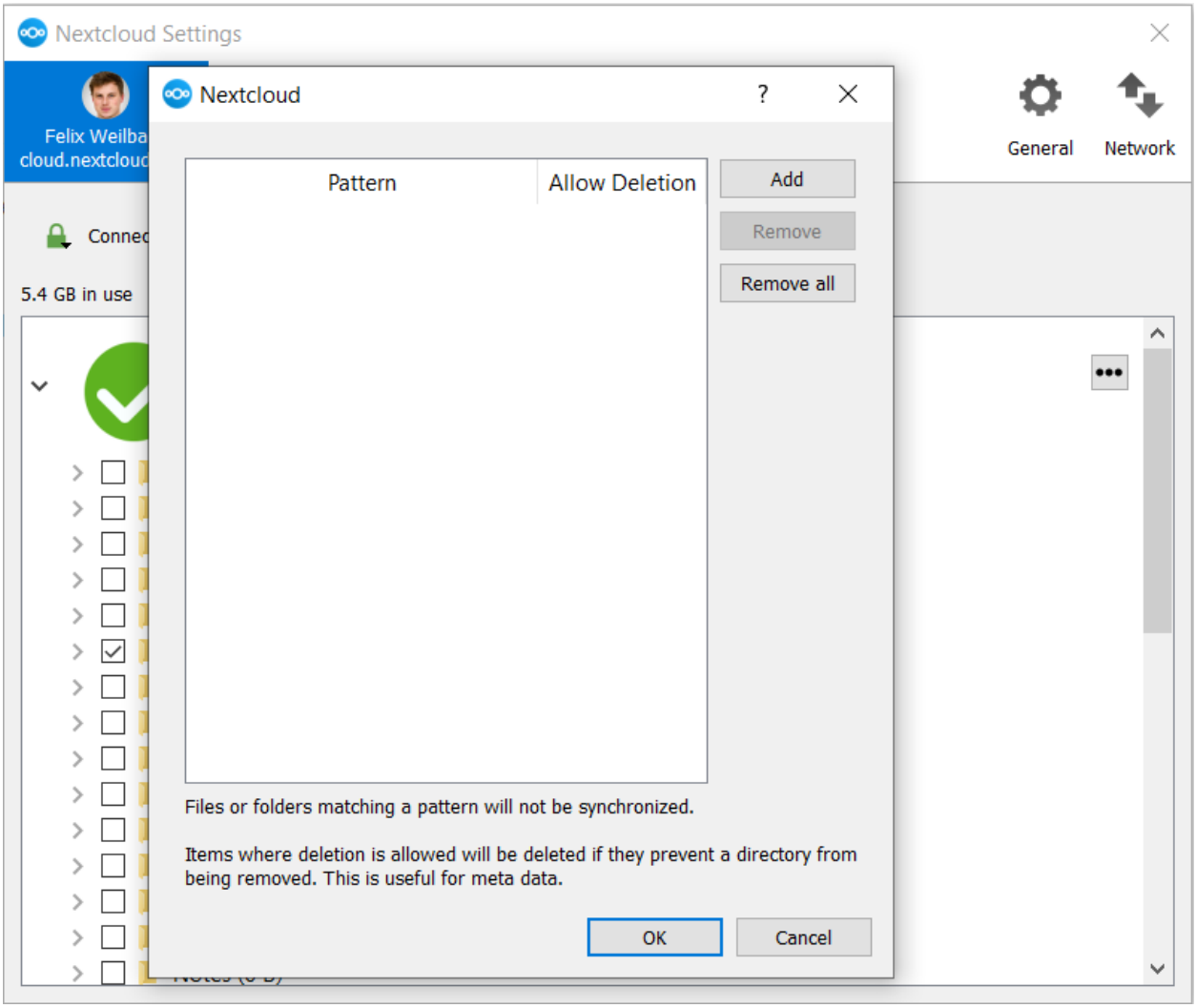
4.6 Using the Network Window

The Network settings window enables you to define network proxy settings, and also to limit download and upload bandwidth.


Nextcloud Settings
Felix Weilbach
cloud.nextcloud.com
General
Network
Proxy Settings
No Proxy
Use system proxy
Specify proxy manually as
HTTP(S) proxy
Host
Hostname of proxy server
8080
Proxy server requires authentication
Username for proxy server
Password for proxy server
Download Bandwidth
No limit
Limit automatically
Limit to
80
KBytes/s
Upload Bandwidth
No limit
Limit automatically
Limit to
10
KBytes/s

4.7 Using the Ignored Files Editor

You might have some local files or directories that you do not want to backup and store on the server. To identify and exclude these files or directories, you can use the *Ignored Files Editor* (General tab.)



For your convenience, the editor is pre-populated with a default list of typical ignore patterns. These patterns are contained in a system file (typically `sync-exclude.lst`) located in the Nextcloud Client application directory. You cannot modify these pre-populated patterns directly from the editor. However, if necessary, you can hover over any pattern in the list to show the path and filename associated with that pattern, locate the file, and edit the `sync-exclude.lst` file.

Note: Modifying the global exclude definition file might render the client unusable or result in undesired behavior.

Each line in the editor contains an ignore pattern string. When creating custom patterns, in addition to being able to use normal characters to define an ignore pattern, you can use wildcards characters for matching values. As an example, you can use an asterisk (*) to identify an arbitrary number of characters or a question mark (?) to identify a single character.

Patterns that end with a slash character (/) are applied to only directory components of the path being checked.

Note: Custom entries are currently not validated for syntactical correctness by the editor, so you will not see any warnings for bad syntax. If your synchronization does not work as you expected, check your syntax.

Each pattern string in the list is followed by a checkbox. When the check box contains a check mark, in addition to ignoring the file or directory component matched by the pattern, any matched files are also deemed "fleeting metadata" and removed by the client.

In addition to excluding files and directories that use patterns defined in this list:

- The Nextcloud Client always excludes files containing characters that cannot be synchronized to other file systems.
- Files are removed that cause individual errors three times during a synchronization. However, the client provides the option of retrying a synchronization three additional times on files that produce errors.

For more detailed information see *Ignored Files*.

CHAPTER

FIVE

CONFLICTS

5.1 Overview

The Nextcloud desktop client uploads local changes and downloads remote changes. When a file has changed on the local side and on the remote between synchronization runs the client will be unable to resolve the situation on its own. It will create a conflict file with the local version, download the remote version and notify the user that a conflict occurred which needs attention.

5.2 Example

Imagine there is a file called `mydata.txt` your synchronized folder. It has not changed for a while and contains the text "contents" locally and remotely. Now, nearly at the same time you update it locally to say "local contents" while the file on the server gets updated to contain "remote contents" by someone else.

When attempting to upload your local changes the desktop client will notice that the server version has also changed. It creates a conflict and you will now have two files on your local machine:

- `mydata.txt` containing "remote contents"
- `mydata (conflicted copy 2018-04-10 093612).txt` containing "local contents"

In this situation the file `mydata.txt` has the remote changes (and will continue to be updated with further remote changes when they happen), but your local adjustments have not been sent to the server (unless the server enables conflict uploading, see below).

The desktop client notifies you of this situation via system notifications, the system tray icon and a yellow "unresolved conflicts" badge in the account settings window. Clicking this badge shows a list that includes the unresolved conflicts and clicking one of them opens an explorer window pointing at the relevant file.

To resolve this conflict, open both files, compare the differences and copy your local changes from the "conflicted copy" file into the base file where applicable. In this example you might change `mydata.txt` to say "local and remote contents" and delete the file with "conflicted copy" in its name. With that, the conflict is resolved.

5.3 Uploading conflicts (experimental)

By default the conflict file (the file with "conflicted copy" in its name that contains your local conflicting changes) is not uploaded to the server. The idea is that you, the author of the changes, are the best person for resolving the conflict and showing the conflict to other users might create confusion.

However, in some scenarios it makes a lot of sense to upload these conflicting changes such that local work can become visible even if the conflict won't be resolved immediately.

In the future there might be a server-wide switch for this behavior. For now it can already be tested by setting the environment variable `OWNCLOUD_UPLOAD_CONFLICT_FILES=1`.

CHAPTER

SIX

ADVANCED USAGE

6.1 Options

You have the option of starting your Nextcloud desktop client with the `nextcloud` command. The following options are supported:

`nextcloud -h` or `nextcloud --help`
Displays all command options.

The other options are:

`--logwindow`
Opens a window displaying log output.

`--logfile` <*filename*>
Write log output to the file specified. To write to stdout, specify - as the filename.

`--logdir` <*name*>
Writes each synchronization log output in a new file in the specified directory.

`--logexpire` <*hours*>
Removes logs older than the value specified (in hours). This command is used with `--logdir`.

`--logflush`
Clears (flushes) the log file after each write action.

`--logdebug`
Also output debug-level messages in the log (equivalent to setting the env var QT_LOGGING_RULES="qt.*=true;*.debug=true").

`--confdir` <*dirname*>
Uses the specified configuration directory.

`--background`
Launch the application in the background (i.e. without opening the main dialog).

6.2 Mass Deployment And Account Creation

It is possible to perform mass deployment of the Nextcloud desktop client by passing certain command-line parameters from the deployment step to Nextcloud desktop client executable after the initial setup. This will allow desktop client to generate a config (.cfg) file that will be used during subsequent launches. A config file will have a corresponding account written into it similar to if you have added it manually via the desktop client's UI. The desktop client will exit with code 0 if account has been added successfully, or 1 in case of failure. Detailed failure message is printed to the desktop client logs.

The following parameters are supported:

`--userid`
: (required) userId (username as on the server) to pass when creating an account via command-line.

`--apppassword`
: (required) appPassword to pass when creating an account via command-line (see the `login-flow` section in server documentation on how to generate the app password).

`--localdirpath`
: (optional) path where to create a local sync folder when creating an account via command-line. If skipped, then default local sync folder path (/home/<userid>/Nextcloud<n> for Linux/mac or C:/<userid>/Nextcloud<n> for Windows) will be generated by desktop client.

`--isvfsenabled`
: (optional) whether to set a VFS or non-VFS folder (1 for 'yes' or 0 for 'no') when creating an account via command-line. Default is 0.

`--remotedirpath`
: (optional) path to a remote subfolder when creating an account via command-line. e.g. If the server has folders "/Photos", "/Documents", "/Music" you can pass "/Music" and then this folder will get set up as remote root.

`--serverurl`
: (required) a server URL to use when creating an account via command-line. (NOTE: There is another parameter supported by Nextcloud desktop client `--overrideserverurl` but it SHOULD NOT be used here as it is intended for setup via UI with wizard)

Examples:

- `C:\Program Files\Nextcloud\nextcloud.exe" --userid admin --apppassword Jliy12356785jxnHa2ZCiZ9MX48ncECwDso95Pq3a5HABjY34ZvhZiXrPfpKWUg7aOHAX5 --localdirpath "D:\\Nextcloud-sync-folder" --remotedirpath /Music --serverurl "https://cloud.example.com" --isvfsenabled 1"` - this will create a config file for user admin on the server https://cloud.example.com and set a remote root folder to "Music", the local sync folder will get created with VFS mode.
- For Linux and mac the same example as above will work but `nextcloud.exe path` and `--localdirpath` value should get changed to platform specific format (e.g. `no .exe extension` and `/home/<user folder>` format)

6.3 Configuration File

The Nextcloud Client reads a configuration file. You can locate this configuration file as follows:

On Linux distributions:
`$HOME/.config/Nextcloud/nextcloud.cfg`

On Microsoft Windows systems:
`%APPDATA%\Nextcloud\nextcloud.cfg`

On macOS systems:
`$HOME/Library/Preferences/Nextcloud/nextcloud.cfg`

The configuration file contains settings using the Microsoft Windows .ini file format. You can overwrite changes using the Nextcloud configuration dialog.

Note: Use caution when making changes to the Nextcloud Client configuration file. Incorrect settings can produce unintended results.

Some interesting values that can be set on the configuration file are:

`[Nextcloud]` section		
Variable	Default	Meaning
`remotePollInterval`	`30000`	Specifies the poll time for the remote repository in milliseconds.
`forceSyncInterval`	`7200000`	The duration of no activity after which a synchronization run shall be triggered automatically.
`fullLocalDiscoveryInterval`	`3600000`	The interval after which the next synchronization will perform a full local discovery.
`notificationRefreshInterval`	`300000`	Specifies the default interval of checking for new server notifications in milliseconds.

[General] section		
Variable	Default	Meaning
chunkSize	10000000 (10 MB)	Specifies the chunk size of uploaded files in bytes. The client will dynamically adjust this size within the maximum and minimum bounds (see below).
forceLoginV2	false	If the client should force the new login flow, eventhough some circumstances might need the old flow.
minChunkSize	5000000 (5 MB)	Specifies the minimum chunk size of uploaded files in bytes.
maxChunkSize	5000000000 (5000 MB)	Specifies the maximum chunk size of uploaded files in bytes.
targetChunkUploadDuration	60000 (1 minute)	Target duration in milliseconds for chunk uploads. The client adjusts the chunk size until each chunk upload takes approximately this long. Set to 0 to disable dynamic chunk sizing.
promptDeleteAllFiles	false	If a UI prompt should ask for confirmation if it was detected that all files and folders were deleted.
timeout	300	The timeout for network connections in seconds.
moveToTrash	false	If non-locally deleted files should be moved to trash instead of deleting them completely.
showExperimentalOptions	false	Whether to show experimental options that are still undergoing testing in the user interface. Turning this on does not enable experimental behavior on its own. It does enable user interface options that can be used to opt in to experimental features.
showMainDialogAsNormalWindow	false	Whether the main dialog should be shown as a normal window even if tray icons are available.

[Proxy] section		
Variable	Default	Meaning
host	127.0.0.1	The address of the proxy server.
port	8080	The port were the proxy is listening.
type	2	0 for System Proxy.
		1 for SOCKS5 Proxy.
		2 for No Proxy.
		3 for HTTP(S) Proxy.

6.4 Environment Variables

The behavior of the client can also be controlled using environment variables. The value of the environment variables overrides the values in the configuration file.

The environment variables are:

- *OWNCLOUD_CHUNK_SIZE* (default: 5242880; 5 MiB) – Specifies the chunk size of uploaded files in bytes. Increasing this value may help with synchronization problems in certain configurations.
- *OWNCLOUD_TIMEOUT* (default: 300 s) – The timeout for network connections in seconds.
- *OWNCLOUD_CRITICAL_FREE_SPACE_BYTES* (default: 50*1000*1000 bytes) - The minimum disk space needed for operation. A fatal error is raised if less free space is available.
- *OWNCLOUD_FREE_SPACE_BYTES* (default: 250*1000*1000 bytes) - Downloads that would reduce the free space below this value are skipped. More information available under the "Low Disk Space" section.
- *OWNCLOUD_MAX_PARALLEL* (default: 6) - Maximum number of parallel jobs.

- *OWNCLOUD_BLACKLIST_TIME_MIN* (default: 25 s) - Minimum timeout for blacklisted files.
- *OWNCLOUD_BLACKLIST_TIME_MAX* (default: 24*60*60 s; one day) - Maximum timeout for blacklisted files.

6.5 Nextcloud Command Line Client

The Nextcloud Client packages contain a command line client, `nextcloudcmd`, that can be used to synchronize Nextcloud files to client machines.

`nextcloudcmd` performs a single *sync run* and then exits the synchronization process. In this manner, `nextcloudcmd` processes the differences between client and server directories and propagates the files to bring both repositories to the same state. Contrary to the GUI-based client, `nextcloudcmd` does not repeat synchronizations on its own. It also does not monitor for file system changes.

6.5.1 Install `nextcloudcmd`

CentOS

```
$ sudo yum -y install epel-release
$ sudo yum -y install nextcloud-client
```

Ubuntu

```
$ sudo add-apt-repository ppa:nextcloud-devs/client
$ sudo apt update
$ sudo apt install nextcloud-client
```

Debian

```
$ sudo apt install nextcloud-desktop-cmd
```

Refer to the link

- https://nextcloud.com/install/#install-clients
- https://launchpad.net/~nextcloud-devs/+archive/ubuntu/client
- https://pkgs.alpinelinux.org/packages?name=nextcloud-client
- https://help.nextcloud.com/t/linux-packages-status/10216

To invoke `nextcloudcmd`, you must provide the local and the remote repository URL using the following command:

```
nextcloudcmd [OPTIONS...] sourcedir nextcloudurl
```

where `sourcedir` is the local directory and `nextcloudurl` is the server URL.

Other command line switches supported by `nextcloudcmd` include the following:

`--path`
: Overrides default remote root folder to a specific subfolder on the server(e.g.: /Documents would sync the Documents subfolder on the server)

`--user, -u [user]`
: Use `user` as the login name.

--password, -p [password]
Use `password` as the password.

-n
Use `netrc (5)` for login.

--non-interactive
Do not prompt for questions.

--silent, --s
Inhibits verbose log output.

--trust
Trust any SSL certificate, including invalid ones.

--httpproxy http://[user@pass:]<server>:<port>
Uses `server` as HTTP proxy.

--exclude [file]
Exclude list file

--unsyncedfolders [file]
File containing the list of un-synced remote folders (selective sync)

--max-sync-retries [n]
Retries maximum n times (defaults to 3)

-h
Sync hidden files, do not ignore them

6.5.2 Credential Handling

`nextcloudcmd` requires the user to specify the username and password using the standard URL pattern, e.g.,

```
$ nextcloudcmd /home/user/my_sync_folder https://carla:secret@server/nextcloud
```

To synchronize the Nextcloud directory `Music` to the local directory `media/music`, through a proxy listening on port `8080`, and on a gateway machine using IP address `192.168.178.1`, the command line would be:

```
$ nextcloudcmd --httpproxy http://192.168.178.1:8080 --path /Music \
               $HOME/media/music \
               https://server/nextcloud
```

`nextcloudcmd` will prompt for the user name and password, unless they have been specified on the command line or `-n` has been passed.

6.5.3 Exclude List

`nextcloudcmd` requires access to an exclude list file. It must either be installed along with `nextcloudcmd` and thus be available in a system location, be placed next to the binary as `sync-exclude.lst` or be explicitly specified with the `--exclude` switch.

6.5.4 Example

- Synchronize a local directory to the specified directory of the nextcloud server

```
$ nextcloudcmd --path /<Directory_that_has_been_created> /home/user/<my_sync_folder> \
https://<username>:<secret>@<server_address>
```

6.6 Low Disk Space

When disk space is low the Nextcloud Client will be unable to synchronize all files. This section describes its behavior in a low disk space situation as well as the options that influence it.

1. Synchronization of a folder aborts entirely if the remaining disk space falls below 50 MB. This threshold can be adjusted with the `OWNCLOUD_CRITICAL_FREE_SPACE_BYTES` environment variable.
2. Downloads that would reduce the free disk space below 250 MB will be skipped or aborted. The download will be retried regularly and other synchronization is unaffected. This threshold can be adjusted with the `OWNCLOUD_FREE_SPACE_BYTES` environment variable.

6.7 Wizard Account Setup Command-line Options

If you want to automate an Account Setup Wizard to allow the user skip entering server URL and local sync folder path in UI, you can use command-line parameters. When you specify both, the desktop client's Account Setup Wizard will jump straight to opening a browser for account authentication/connection without the need of entering any of the connection details. The local sync folder will also be selected to the one you specify instead of using default path (/home/Nextcloud)

The following parameters are supported:

`--overridelocaldir`
: specify a local dir to be used in the account setup wizard (e.g.: /home/nextcloud-sync-folder)

`--overrideserverurl`
: specify a server URL to use for the force override to be used in the account setup wizard (e.g.: https://cloud.example.com)

Examples:

- `C:\Program Files\Nextcloud\nextcloud.exe" --overridelocaldir "D:/work/nextcloud-sync-folder" --overrideserverurl https://cloud.example.com`
- For Linux and mac the same example as above will work but `nextcloud.exe path` and `--overridelocaldir` value should get changed to platform specific format (e.g. `no .exe extension` and `/home/<user folder>` format)

CHAPTER

SEVEN

THE AUTOMATIC UPDATER

The Automatic Updater ensures that you always have the latest features and bug fixes for your Nextcloud synchronization client.

The Automatic Updater updates only on macOS and Windows computers; Linux users only need to use their normal package managers. However, on Linux systems the Updater will check for updates and notify you when a new version is available.

7.1 Basic Workflow

The following sections describe how to use the Automatic Updater on different operating systems.

7.1.1 Windows

The Nextcloud client checks for updates and downloads them when available. You can view the update status under `Settings -> General -> Updates` in the Nextcloud client.

If an update is available, and has been successfully downloaded, the Nextcloud client starts a silent update prior to its next launch and then restarts itself. Should the silent update fail, the client offers a manual download.

Note: Administrative privileges are required to perform the update.

7.1.2 macOS

The macOS client has an autoupdater which uses the Sparkle framework. This autoupdater is bundled into the client App Bundle and checks for updates on launch, notifying you if an update is available. This will present a pop-up that can let you automatically download and install the latest client update with one click.

In versions of the client where the Sparkle-based autoupdater is not bundled, a clickable notification will appear informing of an update being available. Upon clicking on said notification, the download page for the latest version of the client will be opened in the system's web browser.

Like on other systems, you can view the update status under `Settings -> General -> Updates` in the Nextcloud client.

7.1.3 Linux

Linux distributions provide their own update tools, so Nextcloud clients that use the Linux operating system do not perform any updates on their own. The client will inform you (`Settings -> General -> Updates`) when an update is available.

7.2 Preventing Automatic Updates

In controlled environments, such as companies or universities, you might not want to enable the auto-update mechanism, as it interferes with controlled deployment tools and policies. To address this case, it is possible to disable the auto-updater entirely. The following sections describe how to disable the auto-update mechanism for different operating systems.

7.2.1 Preventing Automatic Updates in Windows Environments

Users may disable automatic updates by adding this line to the [General] section of their `nextcloud.cfg` files:

```
skipUpdateCheck=true
```

Windows administrators have more options for preventing automatic updates in Windows environments by using one of two methods. The first method allows users to override the automatic update check mechanism, whereas the second method prevents any manual overrides.

To prevent automatic updates, but allow manual overrides:

1. Edit these Registry keys:
   a. (32-bit-Windows) `HKEY_LOCAL_MACHINE\Software\Nextcloud\Nextcloud`
   b. (64-bit-Windows) `HKEY_LOCAL_MACHINE\Software\Wow6432Node\Nextcloud\Nextcloud`
2. Add the key `skipUpdateCheck` (of type DWORD).
3. Specify a value of 1 to the machine.

To manually override this key, use the same value in `HKEY_CURRENT_USER`.

To prevent automatic updates and disallow manual overrides:

Note: This is the preferred method of controlling the updater behavior using Group Policies.

---

1. Edit this Registry key:

   `HKEY_LOCAL_MACHINE\Software\Policies\Nextcloud GmbH\Nextcloud`

2. Add the key `skipUpdateCheck` (of type DWORD).
3. Specify a value of 1 to the machine.

Note: branded clients have different key names

7.2.2 Preventing Automatic Updates in Linux Environments

Because the Linux client does not provide automatic updating functionality, there is no need to remove the automatic-update check. However, if you want to disable it edit your desktop client configuration file, `$HOME/.config/Nextcloud/nextcloud.cfg`. Add this line to the [General] section:

```
skipUpdateCheck=true
```

CHAPTER

EIGHT

APPENDIX A: BUILDING THE CLIENT

The goal of this section is to set up a build environment for developing and testing the Nextcloud Desktop client. If you just want to use the Nextcloud Desktop client without developing and testing it, you should download the latest stable build instead.

Note: These instructions represent a particular streamlined and easy-to-understand methodology, but they are by no means the only way of setting up a build environment.

The steps listed here have been tested multiple times and should allow you to build the client and/or the documentation with not warnings or errors. These instructions should be current with the version, 3.14, of the Nextcloud Client with which it ships. If you are using the most recent version of these instructions, and you run into errors or warnings with the latest code from the repository, please open a GitHub Issue to let us know so we can document a workaround or fix any underlying problems.

8.1 Using GitHub

By default, cloning the GitHub repository will give you the "master" branch, which is the most recent. If for some reason you want to build an older version of the Nextcloud Desktop client, you can choose a branch corresponding with that version. However, for older versions of the client, please be mindful that any issues present may have been fixed in more recent versions.

Note: Doing anything other than just downloading the existing code will require you to have a GitHub account.

If your goal in cloning and building the Nextcloud Desktop client is to contribute to its development, and you are not already a "collaborator" on the Nextcloud Desktop GitHub repository, you will need to create a "fork" by clicking the "fork" button in the upper right on any GitHub page in the repository. It is important to do this in advance because the URL for cloning the repository is different for a fork than for the main official version.

When cloning a GitHub repository, you have two options for authenticating your GitHub account, SSH or HTTPS. SSH requires additional setup but is more secure and simplifies things later on. For an explanation of the differences between HTTPS and SSH, as well as instructions to set up SSH, see this GitHub help article on the subject.

The most basic version of the Git command for cloning a repository is as follows:

```
$ git clone <repository_url>
```

Which will clone the repository into the directory where you run the command.

The four versions of the `git clone` command are as follows:

1. HTTPS from the official repository:

```
$ git clone https://github.com/nextcloud/desktop.git
```

2. SSH from the official repository:

```
$ git clone git@github.com:nextcloud/desktop.git
```

3. HTTPS from a fork (see above):

```
% git clone https://github.com/<github_username>/desktop.git
```

4. SSH from a fork (see above):

```
% git clone git@github.com:<github_username>/desktop.git
```

8.2 macOS Development Build

Note: While it is possible to do many of the following steps using GUI frontends, wherever possible the Terminal commands are listed instead, in order to streamline the process.

1. Install Xcode from the Mac App Store:

   https://apps.apple.com/app/xcode/id497799835

Then, in Terminal:

2. Install Xcode command line tools:

```
% xcode-select -install
```

3. Install Homebrew from brew.sh (which will just give you the following):

```
% /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/
↪install/HEAD/install.sh)"
```

Note: Under certain circumstances, you may get on error along the lines of `Permission denied @ apply2files` when installing certain Homebrew packages. This is a known issue and can be fixed by changing the permissions on the affected files with the following command:

```
% sudo chown -R $(whoami):admin /usr/local/* \
   && sudo chmod -R g+rwx /usr/local/*
```

This workaround may lead to other shell warnings.

4. Install Homebrew packages:

```
% brew install git qt qtkeychain cmake openssl glib cmocka karchive
```

5. Certain Homebrew packages are not automatically linked in places where the build scripts can find them, so you can create a shell-profile script that will find and load them dynamically when you run a build:

```
% echo 'export QT_PATH=$(brew --prefix qt6)/bin' >> ~/.nextcloud_build_variables
% echo 'export CMAKE_PREFIX_PATH=$(brew --prefix qt6);$(brew --prefix karchive)' >>␣
↪~/.nextcloud_build_variables
```

Note: The name `~/.nextcloud_build_variables` is just a suggestion for convenience. You can use a different file or create an entire shell script, but this way of doing things is the simplest to explain.

6. Clone the Nextcloud repository to a convenient location, such as `~/Repositories`:

```
% mkdir ~/Repositories
```

(if it doesn't already exist), then:

```
% cd ~/Repositories
```

Note: The cloned repository can go basically anywhere your user account has write access, though it should not go in a directory synced with another cloud service (especially not iCloud Drive). `~/Repositories` is recommended for tidiness and consistency.

```
% git clone <repository_url>
```

(See the above section on using GitHub for an explanation of what URL to use.)

7. Create build directory:

```
% cd ~/Repositories/desktop
% mkdir build
```

8. Generate the build files:

Note:

By default Nextcloud Desktop will build in a protected directory on macOS, so you need to specify a build location. You can do this every time you build, or you can add it to your save build variables, like so:

```
% echo 'export CMAKE_INSTALL_PREFIX=~/Builds' >> ~/.nextcloud_build_variables
# If you want to build a macOS app bundle for distribution
% echo 'export BUILD_OWNCLOUD_OSX_BUNDLE=ON' >> ~/.nextcloud_build_variables
```

Replace `~/Builds` with a different directory if you'd like the build to end up elsewhere.

```
% source ~/.nextcloud_build_variables
% cd ~/Repositories/desktop/build
% cmake ..
```

9. Compile and install:

```
% make install
```

8.3 Windows Development Build

8.4 System requirements

- Windows 10 or Windows 11
- The desktop client code
- Python 3
- PowerShell
- Microsoft Visual Studio 2022 and tools to compile C++
- KDE Craft

8.5 Setting up Microsoft Visual Studio

1. Click on 'Modify' in the Visual Studio Installer:
2. Select 'Desktop development with C++'

8.6 Handling the dependencies

We handle the dependencies using KDE Craft because it is easy to set it up and it makes the maintenance much more reliable in all platforms.

1. Set up KDE Craft as instructed in Get Involved/development/Windows - KDE Community Wiki - it requires Python 3 and PowerShell.
2. After running:

```
C:\CraftRoot\craft\craftenv.ps1
```

3. Add the desktop client blueprints - the instructions to handle the client dependencies:

```
craft --add-blueprint-repository [git]https://github.com/nextcloud/desktop-client-
↪blueprints.git
craft craft
```

4. Install all client dependencies:

```
craft --install-deps nextcloud-client
```

8.7 Compiling

1. Make sure your environment variable %PATH% has no conflicting information to the environment you will use to compile the client. For instance, if you have installed OpenSSL previously and have added it to %PATH%, the OpenSSL installed might be a different version than what was installed via KDE Craft.
2. Open the Command Prompt (cmd.exe)
3. Run:

```
"C:\Program Files\Microsoft Visual Studio\2022\Community\VC\Auxiliary\Build\vcvarsall.bat
↪" x64
```

4. To use the tools installed with Visual Studio, you need the following in your %PATH%:
5. Alternatively you can use the tools installed with KDE Craft by adding them to %PATH%:

```
set "PATH=C:\CraftRoot\bin;C:\CraftRoot\dev-utils\bin;%PATH%"
```

Note: C:CraftRoot is the path used by default by KDE Craft. When you are setting it up you may choose a different folder.

6. Create build folder, run cmake, compile and install:

```
cd <desktop-repo-path>
mkdir build
cd build
cmake .. -G Ninja -DCMAKE_INSTALL_PREFIX=. -DCMAKE_PREFIX_PATH=C:\CraftRoot -DCMAKE_
↪BUILD_TYPE=RelWithDebInfo
cmake --build . --target install
```

7. Now you can use Qt Creator to import the build folder with its configurations to be able to work with the code.

8.8 Windows Installer (i.e. Deployment) Build (Cross-Compile)

Due to the large number of dependencies, building the client installer for Windows is **currently only officially supported on openSUSE**, by using the MinGW cross compiler. You can set up any currently supported version of openSUSE in a virtual machine if you do not have it installed already.

In order to make setup simple, you can use the provided Dockerfile to build your own image.

1. Assuming you are in the root of the Nextcloud Client's source tree, you can build an image from this Dockerfile like this:

   ```
   cd admin/win/docker
   docker build . -t nextcloud-client-win32:<version>
   ```

   Replace `<version>` by the version of the client you are building, e.g. 3.14 for the release of the client that this document describes. If you do not wish to use docker, you can run the commands in `RUN` manually in a shell, e.g. to create your own build environment in a virtual machine.

   Note: Docker images are specific to releases. This one refers to 3.14. Newer releases may have different dependencies, and thus require a later version of the docker image! Always pick the docker image fitting your

release of Nextcloud client!

2. From within the source tree Run the docker instance:

```
docker run -v "$PWD:/home/user/client" nextcloud-client-win32:<version> \
   /home/user/client/admin/win/docker/build.sh client/  $(id -u)
```

It will run the build, create an NSIS based installer, as well as run tests. You will find the resulting binary in an newly created `build-win32` subfolder.

If you do not wish to use docker, and ran the `RUN` commands above in a virtual machine, you can run the indented commands in the lower section of `build.sh` manually in your source tree.

4. Finally, you should sign the installer to avoid warnings upon installation. This requires a Microsoft Authenticode Certificate `osslsigncode` to sign the installer:

```
osslsigncode -pkcs12 $HOME/.codesign/packages.pfx -h sha256 \
          -pass yourpass \
          -n "ACME Client" \
          -i "http://acme.com" \
          -ts "http://timestamp.server/" \
          -in ${unsigned_file} \
          -out ${installer_file}
```

For `-in`, use the URL to the time stamping server provided by your CA along with the Authenticode certificate. Alternatively, you may use the official Microsoft `signtool` utility on Microsoft Windows.

If you're familiar with docker, you can use the version of `osslsigncode` that is part of the docker image.

8.9 Generic Build Instructions

Compared to previous versions, building the desktop sync client has become easier. Unlike earlier versions, CSync, which is the sync engine library of the client, is now part of the client source repository and not a separate module.

To build the most up-to-date version of the client:

1. Clone the latest versions of the client from Git as follows:

```
$ git clone git://github.com/nextcloud/client.git
$ cd client
$ git submodule update --init
```

2. Create the build directory

```
$ mkdir client-build
$ cd client-build
```

3. Configure the client build

```
$ cmake -DCMAKE_BUILD_TYPE="Debug" ..
```

Note: You must use absolute paths for the `include` and `library` directories.

Note: On macOS, you need to specify `-DCMAKE_INSTALL_PREFIX=target`, where `target` is a private location, i.e. in parallel to your build dir by specifying `../install`.

Note: qtkeychain must be compiled with the same prefix e.g `CMAKE_INSTALL_PREFIX=/Users/path/to/client/install/ .`

Note: Example:: `cmake -DCMAKE_PREFIX_PATH=/usr/local/opt/qt6 -DCMAKE_INSTALL_PREFIX=/Users/path/to/client/install/`

4. Call `make`.

   The Nextcloud binary will appear in the `bin` directory.

5. (Optional) Call `make install` to install the client to the `/usr/local/bin` directory.

The following are known cmake parameters:

- **`QTKEYCHAIN_LIBRARY=/path/to/qtkeychain.dylib -DQTKEYCHAIN_INCLUDE_DIR=/path/to/qtkeychain/:`**
  Used for stored credentials. When compiling with Qt5, the library is called `qt5keychain.dylib`. You need to compile QtKeychain with the same Qt version.
- `WITH_DOC=TRUE`: Creates doc and manpages through running `make`; also adds install statements, providing the ability to install using `make install`.
- `CMAKE_PREFIX_PATH=/path/to/Qt6/6.7.0/yourarch/lib/cmake/`: Builds using Qt6.
- `CMAKE_INSTALL_PREFIX=path`: Set an install prefix. This is mandatory on Mac OS

8.9.1 Address Sanitizer

You can enable the address sanitizer to detect memory corruptions and other mistakes. The are the following sanitizers are available:

- Address Sanitizer
- Leak anitizer
- Memory sanitizer
- Undefined sanitizer
- Threads sanitizer

You can enable one or more sanitizers through CMake. For example, to enable the address and the undefined sanitizer, execute CMake like `cmake .. -D ECM_ENABLE_SANITIZERS="address;undefined"`. Keep in mind that not all combinations of sanitizers work together, and on some platforms, not all types of sanitizers are available. For example, on Windows there is currently only the address sanitizer available. If you are on Windows, you need to make sure that the linker can find the sanitizer dlls at runtime. If you installed Visual Studio in the standard location, you could find them in **C:/ProgramFiles (x86)/Microsoft Visual Studio/2019/Community/VC/Tools/Llvm/x64/lib/clang/10.0.0/lib/windows**. Make sure you add this location to your path. You may also need to upgrade your Visual Studio version.

Note: If you use Visual Studio on Windows, you can enable the sanitizer if you click on **Manage Configurations**, scroll down to the section **CMake Command Arguments** and enter then `-D ECM_ENABLE_SANITIZERS="address"`

in the text input field below. After that, click on **Save and generate CMake cache to load variables** right above the table.

CHAPTER

NINE

APPENDIX B: HISTORY AND ARCHITECTURE

Nextcloud provides desktop sync clients to synchronize the contents of local directories from computers, tablets, and handheld devices to the Nextcloud server.

Synchronization is accomplished using csync, a bidirectional file synchronizing tool that provides both a command line client as well as a library. A special module for csync was written to synchronize with the Nextcloud built-in WebDAV server.

The Nextcloud Client software is written in C++ using the Qt Framework. As a result, the Nextcloud Client runs on Linux, Windows, and MacOS.

9.1 The Synchronization Process

The process of synchronization keeps files in two separate repositories the same. When synchronized:

- If a file is added to one repository it is copied to the other synchronized repository.
- When a file is changed in one repository, the change is propagated to any other synchronized repository.
- If a file is deleted in one repository, it is deleted in any other.

It is important to note that the Nextcloud synchronization process does not use a typical client/server system where the server is always master. This is a major difference between the Nextcloud synchronization process and other systems like a file backup, where only changes to files or folders and the addition of new files are propagated, but these files and folders are never deleted unless explicitly deleted in the backup.

During synchronization, the Nextcloud Client checks both repositories for changes frequently. This process is referred to as a *sync run*. In between sync runs, the local repository is monitored by a file system monitoring process that starts a sync run immediately if something was edited, added, or removed.

9.2 Synchronization by Time versus ETag

Until the release of the client version 1.1, the Nextcloud synchronization process employed a single file property – the file modification time – to decide which file was newer and needed to be synchronized to the other repository.

The *modification timestamp* is part of the files metadata. It is available on every relevant filesystem and is the typical indicator for a file change. Modification timestamps do not require special action to create, and have a general meaning. One design goal of csync is to not require a special server component. This design goal is why csync was chosen as the backend component.

To compare the modification times of two files from different systems, csync must operate on the same base. Before client version 1.1.0, csync required both device repositories to run on the exact same time. This requirement was achieved through the use of enterprise standard NTP time synchronization on all machines.

Because this timing strategy is rather fragile without the use of NTP, the Nextcloud server provides a unique number that changes whenever the file changes. Although this number is a unique value, it is not a hash of the file. Instead, it is a randomly chosen number, that is transmitted in the Etag field. Because the file number changes if the file changes, its use is guaranteed to determine if one of the files has changed and, thereby, launching a synchronization process.

Before the 1.3.0 release of the Desktop Client, the synchronization process might create false conflict files if time deviates. Original and changed files conflict only in their timestamp, but not in their content. This behavior was changed to employ a binary check if files differ.

Like files, directories also hold a unique ID that changes whenever one of the contained files or directories is modified. Because this is a recursive process, it significantly reduces the effort required for a synchronization cycle, because the client only analyzes directories with a modified ID.

9.3 Comparison and Conflict Cases

As mentioned above, during a *sync run* the client must first detect if one of the two repositories have changed files. On the local repository, the client traverses the file tree and compares the modification time of each file with an expected value stored in its database. If the value is not the same, the client determines that the file has been modified in the local repository.

Note: On the local side, the modification time is a good attribute to use for detecting changes, because the value does not depend on time shifts and such.

For the remote (that is, Nextcloud server) repository, the client compares the ETag of each file with its expected value. Again, the expected ETag value is queried from the client database. If the ETag is the same, the file has not changed and no synchronization occurs.

In the event a file has changed on both the local and the remote repository since the last sync run, it can not easily be decided which version of the file is the one that should be used. However, changes to any side will not be lost. Instead, a *conflict case* is created. The client resolves this conflict by renaming the local file, appending a conflict label and timestamp, and saving the remote file under the original file name.

Example: Assume there is a conflict in message.txt because its contents have changed both locally and remotely since the last sync run. The local file with the local changes will be renamed to message_conflict-20160101-153110.txt and the remote file will be downloaded and saved as message.txt.

Conflict files are always created on the client and never on the server.

9.4 Ignored Files

The Nextcloud Client supports the ability to exclude or ignore certain files from the synchronization process. Some system wide file patterns that are used to exclude or ignore files are included with the client by default and the Nextcloud Client provides the ability to add custom patterns.

By default, the Nextcloud Client ignores the following files:

- Files matched by one of the patterns defined in the Ignored Files Editor.
- Files starting with `._sync_*.db*`, `.sync_*.db*`, `.csync_journal.db*`, `.owncloudsync.log*`, as these files are reserved for journalling.
- Files with a name longer than 254 characters.
- The file `Desktop.ini` in the root of a synced folder.

- Files matching the pattern `*_conflict-*` unless conflict file uploading is enabled.
- Windows only: Files containing characters that do not work on typical Windows filesystems (`\, /, :, ?, *, ", >, <, |`).
- Windows only: Files with a trailing space or dot.
- Windows only: Filenames that are reserved on Windows.

If a pattern selected using a checkbox in the *ignoredFilesEditor-label* (or if a line in the exclude file starts with the character `]` directly followed by the file pattern), files matching the pattern are considered *fleeting meta data*.

These files are ignored and *removed* by the client if found in the synchronized folder. This is suitable for meta files created by some applications that have no sustainable meaning.

If a pattern ends with the forward slash (`/`) character, only directories are matched. The pattern is only applied for directory components of filenames selected using the checkbox.

To match filenames against the exclude patterns, the UNIX standard C library function `fnmatch` is used. This process checks the filename against the specified pattern using standard shell wildcard pattern matching. For more information, please refer to The opengroup website.

The path that is checked is the relative path under the sync root directory.

Pattern and File Match Examples:

Pattern	File Matches
`~$*`	`~$foo`, `~$example.doc`
`fl?p`	`flip`, `flap`
`moo/`	`map/moo/`, `moo/`

9.5 The Sync Journal

The client stores the ETag number in a per-directory database, called the *journal*. This database is a hidden file contained in the directory to be synchronized.

If the journal database is removed, the Nextcloud Client CSync backend rebuilds the database by comparing the files and their modification times. This process ensures that both server and client are synchronized using the appropriate NTP time before restarting the client following a database removal.

9.6 Custom WebDAV Properties

In the communication between client and server a couple of custom WebDAV properties were introduced. They are either needed for sync functionality or help have a positive effect on synchronization performance.

This chapter describes additional XML elements which the server returns in response to a successful PROPFIND request on a file or directory. The elements are returned in the namespace `oc`.

9.7 Server Side Permissions

The XML element <oc:permissions> represents the permission- and sharing state of the item. It is a list of characters, and each of the chars has a meaning as outlined in the table below:

Code	Resource	Description
S	File or Folder	is shared
R	File or Folder	can share (includes re-share)
M	File or Folder	is mounted (like on Dropbox, Samba, etc.)
W	File	can write file
C	Folder	can create file in folder
K	Folder	can create folder (mkdir)
D	File or Folder	can delete file or folder
N	File or Folder	can rename file or folder
V	File or Folder	can move file or folder

Example:

<oc:permissions>RDNVCK</oc:permissions>

9.8 File- or Directory Size

The XML element <oc:size> represents the file- or directory size in bytes. For directories, the size of the whole file tree underneath the directory is accumulated.

Example:

<oc:size>2429176697</oc:size>

9.9 FileID

The XML element <oc:id> represents the so called file ID. It is a non volatile string id that stays constant as long as the file exists. It is not changed if the file changes or is renamed or moved.

Example:

<oc:id>00000020oc5cfy6qqizm</oc:id>

9.10 End-to-end Encryption

Nextcloud is built around the fundamental assumption that, as you can host your own Nextcloud server, you can trust it with your data. This assumption means data on the Nextcloud server can be provided to users through a browser interface. Users can browse their files online, access their calendars and mail and other data from the respective apps and share and collaboratively edit documents with others including guests and users without an account. While data on the server can be encrypted, this is largely designed to protect it from malicious storage solutions or theft of the whole hardware. System administrators always have access to the data.

But for a subset of data, this assumption of trust might not hold true. For example, at an enterprise, the documents of the Human Resources department or the financial department are too sensitive to allow system administrators who manage the server, access them. As a private user, you might trust your hosting provider with the vast majority of your

data but not with medical records. And even if there is trust in the server administration team, a breach of the server can never entirely be ruled out and for some data, even a tiny risk is unacceptable.

The Nextcloud End-to-end Encryption feature (E2EE) was designed to offer protection against a full compromise of the server. See for more details our blog about the threat model for the encryption solutions in Nextcloud and our webpage about End-to-end Encryption. If the end-to-end encryption app is enabled on the server, users can use one of the clients to select a local folder and enable this feature. This will ensure the client encrypts data before it is transmitted to the server.

The first time E2EE is enabled on a folder in any of the clients, the user is prompted with a private key consisting of 12 security words. The user is strongly recommended to record these somewhere secure as the complete loss of this private key means there is no way to access their data anymore. The key is also securely stored in the device's key storage and can be shown on demand. Making the folder available on a second device requires entering this key. Future versions of Nextcloud clients will be able to display a QR code to simplify the process of adding devices. Sharing with other users will not require any special keys or passwords.

Encrypting files locally means the server has no access to them. This brings with it a number of limitations:

- E2EE files can not be accessed or previewed through the web interface
- E2EE files can not be edited with Online Office solutions
- E2EE files can not be shared with a public link
- E2EE files can not be searched, tagged, commented on and have no versioning or trash bin
- E2EE files can not be accessed in other Nextcloud Apps. This means they have no chat sidebar, can not be attached to emails or deck cards, shared in Talk rooms and so on
- E2EE results in slower syncing of file and works poorly or not at all with large files and large quantities of files

These limitations are fundamental to how securely implemented end-to-end encryption works. We realize there are some solutions that call their technology 'end-to-end encryption' but with browser access. Reality is that offering browser access to end-to-end encrypted files would essentially negate any of the benefits of end-to-end encryption. Serving a file in the browser means the server needs to be able to read the files. But if the server can read the files, administrator or a malicious attacker who gained access to the server, can too. Decrypting the file in the browser does not solve this security risk in the least, as the javascript code that would be needed to decrypt the file comes FROM the server, and of course a compromised server would simply send modified javascript code which sends a copy of the encryption keys to the attacker without anybody noticing. See for more details our blog about the threat model for the encryption solutions in Nextcloud and our webpage about End-to-end Encryption.

The E2EE design of Nextcloud allows for sharing on a per-folder level to individual users (not groups), but, as of early 2021, this feature is still on the road map for implementation in the clients.

Due to all these limitations that are inherent to true end-to-end encryption, it is only recommended for a small subset of files, in just a small number of folders. Encrypting your entire sync folder is likely to result in poor performance and sync errors and if you do not trust your server at all, Nextcloud is perhaps not the right solution for your use case. You might instead want to use encrypted archives or another solution.

Note:

- End-to-end Encryption works with Virtual Files (VFS) but only on a per-folder level. Folders with E2EE have to be made available offline in their entirety to access the files, they can not be retrieved on demand in the folder.

9.11 Virtual Files

Note:

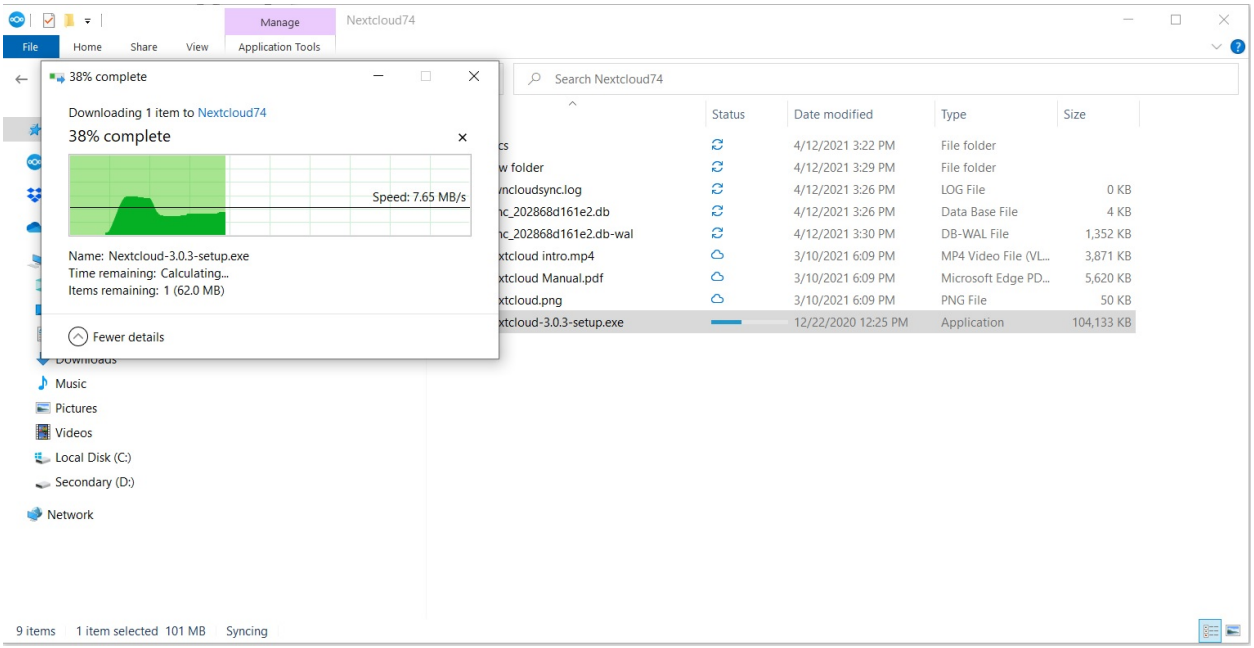
- This feature is currently only available on `Windows` by default. `Linux` and `macOS` implementations are experimental and must be enabled by adding `showExperimentalOptions=true` to the `nextcloud.cfg` configuration file in the `App Data` folder.

Oftentimes, users are working with a huge amount of files that are big in size. Synchronizing every such file to a device that's running a Nextcloud desktop client is not always possible due to the user's device storage space limitation. Let's assume that your desktop client is connected to a server that has 1TB of data. You want all those files at hand, so you can quickly access any file via the file explorer. Your device has 512GB local storage device. Obviously, it's not possible to synchronize even half of 1TB of data that is on the server. What should you do in this case? Of course, you can just utilize the Selective Sync feature, and keep switching between different folders, in such a way that you only synchronize those folders that you are currently working with. Needless to say, this is far from being convenient.

That's why, starting from 3.2.0, we are introducing the VFS (Virtual Files) feature. You may have had experience working with a similar feature in other cloud sync clients. This feature is known by different names: Files On-Demand, SmartSync, etc. The VFS does not occupy much space on the user's storage. It just creates placeholders for each file and folder. These files are quite small and only contain metadata needed to display them properly and to fetch the actual file when needed.

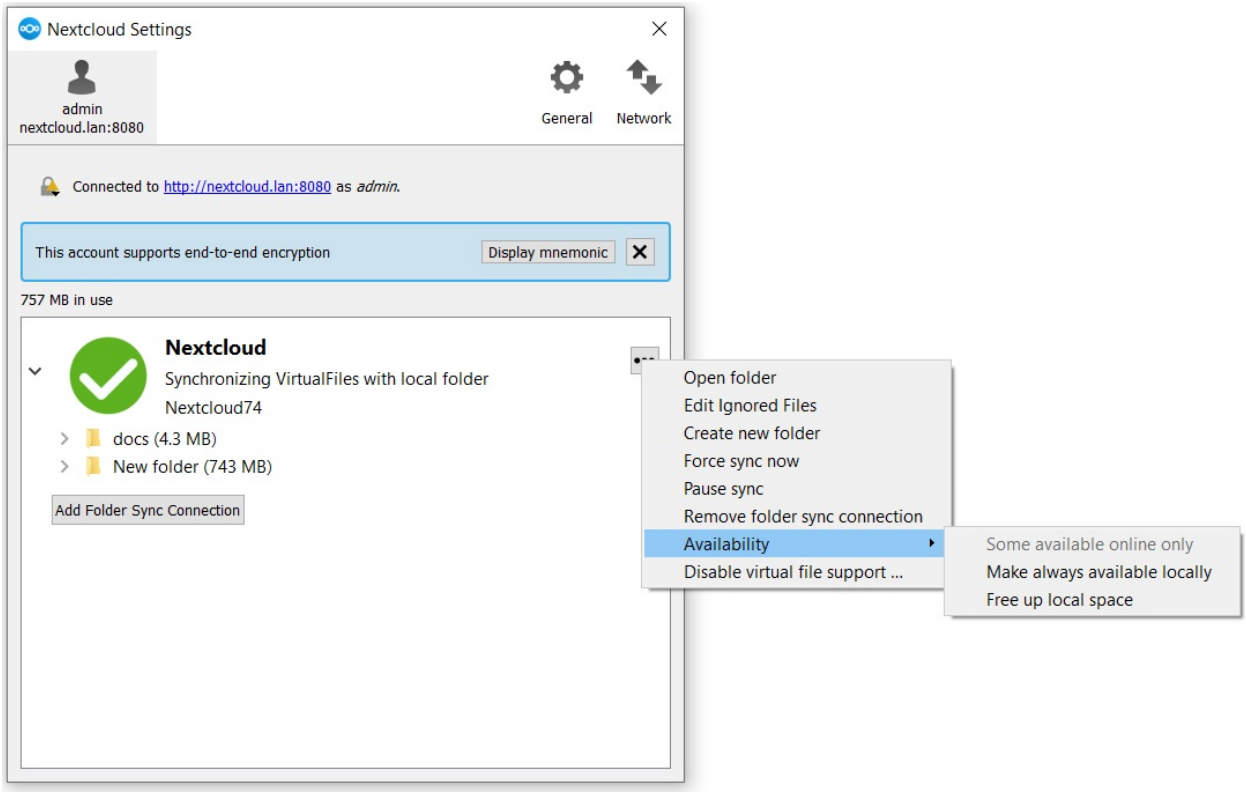
When one tries to open a file, for example by double clicking on a file in the Windows Explorer, one will see that the file gets downloaded and becomes available locally. This can be observed by a small progress-bar popup if the file is large enough.

As soon as the download is complete, the file will then be opened normally as now it is a real file on the user's storage. It won't disappear, and, from now on, will always be available, unless it is manually dehydrated.



As long as the VFS is enabled, a user can choose to remove files that are no longer needed from the local storage. This can be achieved by right-clicking the file/folder in the explorer, and then, choosing "Free up local space" from the context menu. Alternatively, space can be freed up by right-clicking the sync folder in the Settings dialog. It is also possible to make files always hydrated, or, in other words, always available locally. A user just needs to choose the

"Make always available locally" option in the aforementioned context menus.



The VFS can also be disabled if needed, so, the entire folder will then be synced normally. This option is available in the context menu of a sync folder in the Settings dialog. Once disabled, the VFS can also be enabled back by using the same context menu. Files that must be removed from the local storage only, need to be dehydrated via the "Free up local space" option, so, the placeholder will get created in place of real files.

Note:

- End-to-end Encryption works with Virtual Files (VFS) but only on a per-folder level. Folders with E2EE can be made available offline in their entirety, but the individual files in them can not be retrieved on demand. This is mainly due to two technical reasons. First, the Windows VFS API is not designed for handling encrypted files. Second, while the VFS is designed to deal mostly with large files, E2EE is mostly recommended for use with small files as encrypting and decrypting large files puts large demands on the computer infrastructure.

9.12 Local file editing

The Nextcloud desktop GUI client supports local editing when opening a URL that starts with a scheme `nc://` followed by an `open` command, followed by a user email (with port when needed), followed by file path relative to remote root.

Examples of URLs that Nextcloud can handle if the user email and a path to a file is correct: - `nc://open/admin@example.cloud:8080/Photos/lovely.jpg` - `nc://open/user@example.cloud/Photos/lovely.jpg` - `nc://open/user@example.cloud/Documents/sheets/report.xlsx` - `nc://open/user@example.cloud/Documents/docs/document.docx`

Note:

- All the file paths that begin after user email are relative to remote root (/).
- **The server is responsible for generating a correct URL that a user then clicks to edit file locally.**
  - The Nextcloud desktop client is registered in macOS, Linux, and Windows as a custom URI handler for the `nc://` scheme.
  - The URL is parsed and validated by Nextcloud desktop client, so, opening an incorrectly formatted URL will not have any effect.
  - The port after user email is necessary if the default :80 or :443 is not used. The rule of thumb is to always have a port added if you need it when accessing your server via Web UI

CHAPTER

TEN

APPENDIX C: TROUBLESHOOTING

The following two general issues can result in failed synchronization:

- The server setup is incorrect.
- The client contains a bug.

When reporting bugs, it is helpful if you first determine what part of the system is causing the issue.

10.1 Identifying Basic Functionality Problems

Performing a general Nextcloud Server test

The first step in troubleshooting synchronization issues is to verify that you can log on to the Nextcloud web application. To verify connectivity to the Nextcloud server try logging in via your Web browser.

If you are not prompted for your username and password, or if a red warning box appears on the page, your server setup requires modification. Please verify that your server installation is working correctly.

Ensure the WebDAV API is working

If all desktop clients fail to connect to the Nextcloud Server, but access using the Web interface functions properly, the problem is often a misconfiguration of the WebDAV API.

The Nextcloud Client uses the built-in WebDAV access of the server content. Verify that you can log on to Nextcloud's WebDAV server. To verify connectivity with the Nextcloud WebDAV server:

- Open a browser window and enter the address to the Nextcloud WebDAV server.

For example, if your Nextcloud instance is installed at `http://yourserver.com/nextcloud`, your WebDAV server address is `http://yourserver.com/nextcloud/remote.php/dav`.

If you are prompted for your username and password but, after providing the correct credentials, authentication fails, please ensure that your authentication backend is configured properly.

Use a WebDAV command line tool to test

A more sophisticated test method for troubleshooting synchronization issues is to use a WebDAV command line client and log into the Nextcloud WebDAV server. One such command line client – called `cadaver` – is available for Linux distributions. You can use this application to further verify that the WebDAV server is running properly using PROPFIND calls.

As an example, after installing the `cadaver` app, you can issue the `propget` command to obtain various properties pertaining to the current directory and also verify WebDAV server connection.

10.2 "CSync unknown error"

If you see this error message stop your client, delete the `.sync_xxxxxxx.db` file, and then restart your client. There is a hidden `.sync_xxxxxxx.db` file inside the folder of every account configured on your client.

Note: Please note that this will also erase some of your settings about which files to download.

See https://github.com/owncloud/client/issues/5226 for more discussion of this issue.

10.3 "Connection closed" message when syncing files

This message can be caused by using chunks that are too big or time-outs that are set too liberally. You can configure the chunking behavior of the client in the config file. For example, change these settings:

`chunkSize`	`10000000` (10 MB)	Specifies the chunk size of uploaded files in bytes. The client will dynamically adjust this size within the maximum and minimum bounds (see below).
`minChunkSize`	`1000000` (1 MB)	Specifies the minimum chunk size of uploaded files in bytes.
`maxChunkSize`	`50000000` (1000 MB)	Specifies the maximum chunk size of uploaded files in bytes.
`targetChunkUploadDuration`	`60000` (1 minute)	Target duration in milliseconds for chunk uploads. The client adjusts the chunk size until each chunk upload takes approximately this long. Set to 0 to disable dynamic chunk sizing.

Setting `maxChunkSize` to 50000000, for example, will decrease the individual chunk to about 50 mb. This causes additional overhead but might be required in some situations, for example behind CloudFlare which has been seen limiting upload chunks to 100mb. In other situations, limiting `targetChunkUploadDuration` can help to avoid time-outs.

10.4 Isolating other issues

Other issues can affect synchronization of your Nextcloud files:

- If you find that the results of the synchronizations are unreliable, please ensure that the folder to which you are synchronizing is not shared with other synchronization applications.
- Synchronizing the same directory with Nextcloud and other synchronization software such as Unison, rsync, Microsoft Windows Offline Folders, or other cloud services such as Dropbox or Microsoft SkyDrive is not supported and should not be attempted. In the worst case, it is possible that synchronizing folders or files using Nextcloud and other synchronization software or services can result in data loss.
- If you find that only specific files are not synchronized, the synchronization protocol might be having an effect. Some files are automatically ignored because they are system files, other files might be ignored because their filename contains characters that are not supported on certain file systems. For more information about ignored files, see *Ignored Files*.
- If you are operating your own server, and use the local storage backend (the default), make sure that Nextcloud has exclusive access to the directory.

Warning: The data directory on the server is exclusive to Nextcloud and must not be modified manually.

- If you are using a different file backend on the server, you can try to exclude a bug in the backend by reverting to the built-in backend.
- If you are experiencing slow upload/download speed or similar performance issues be aware that those could be caused by on-access virus scanning solutions, either on the server (like the files_antivirus app) or the client.

10.5 Log Files

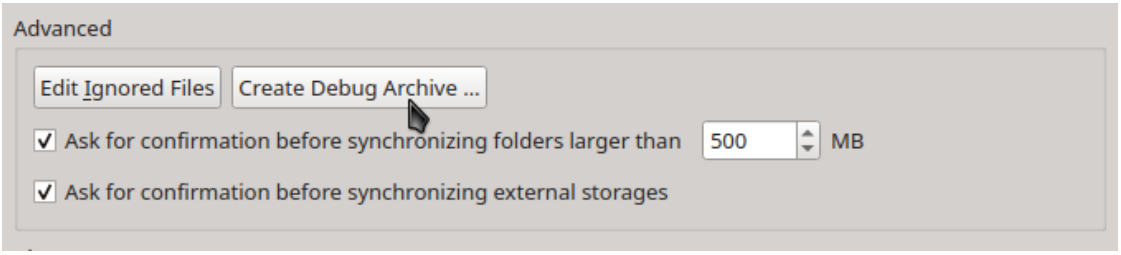
Effectively debugging software requires as much relevant information as can be obtained. To assist the Nextcloud support personnel, please try to provide as many relevant logs as possible. Log output can help with tracking down problems and, if you report a bug, log output can help to resolve an issue more quickly.

Warning: Log files contain sensitive information. You may wish to redact sensitive details or to only share limited excerpts.

10.5.1 Obtaining the Client Log File

10.5.2 Create Debug Archive

Since the 3.1.0 release we made it easier for users to provide debug information: debug logging is enabled by default with expiration time set to 24 hours and under the "General" settings, you can click on "Create Debug Archive ..." to pick the location of where the desktop client will export the logs and the database to a zip file.

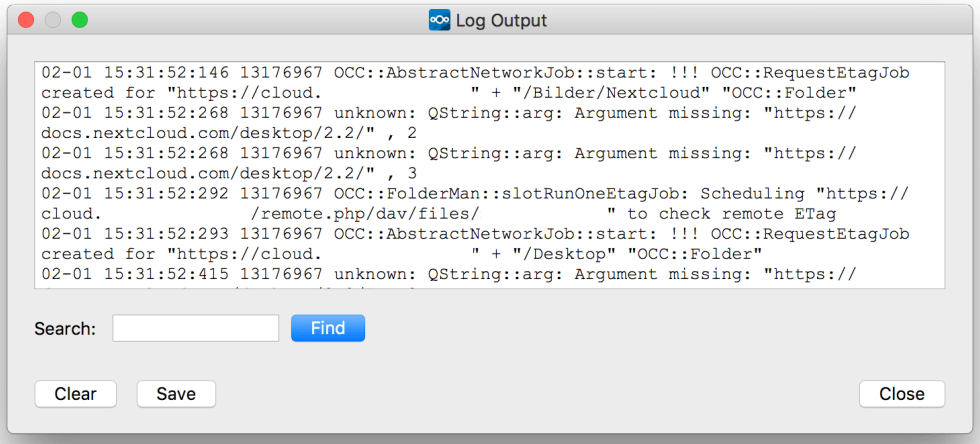


10.5.3 Keyboard shortcut

Another way to obtain the client log file:

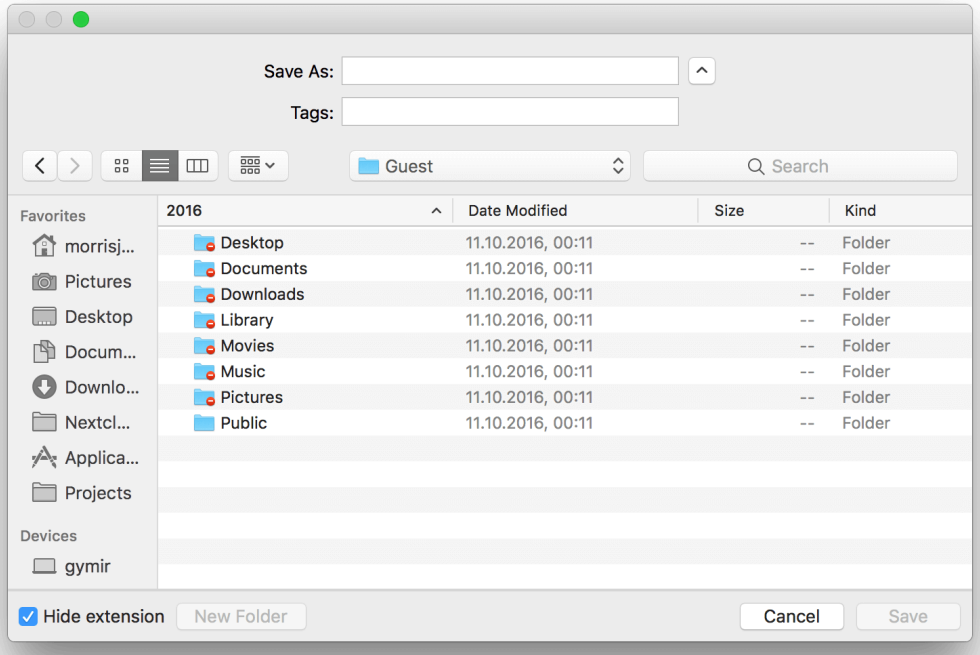
1. Open the Nextcloud Desktop Client.
2. Press F12 or Ctrl-L on your keyboard.

   The Log Output window opens.



3. Click the ‘Save’ button.

   The Save Log File window opens.



4. Migrate to a location on your system where you want to save your log file.
5. Name the log file and click the ‘Save’ button.

   The log file is saved in the location specified.

10.5.4 Command line

Alternatively, you can launch the Nextcloud Log Output window using the `--logwindow` command. After issuing this command, the Log Output window opens to show the current log. You can then follow the same procedures mentioned above to save the log to a file.

Note: You can also open a log window for an already running session, by restarting the client using the following command:

- Windows: `C:\Program Files (x86)\Nextcloud\nextcloud.exe --logwindow`
- macOS: `/Applications/nextcloud.app/Contents/MacOS/nextcloud --logwindow`
- Linux: `nextcloud --logwindow`

10.5.5 Config file

The Nextcloud client enables you to save log files directly to a predefined file or directory. This is a useful option for troubleshooting sporadic issues as it enables you to log large amounts of data and bypass the limited buffer settings associated with the log window.

To enable logging to a directory, stop the client and add the following to the General section in the configuration file:

```
[General]
logDebug=true
logExpire=<hours>
logDir=<dir>
```

Independent of platform you must use slash (/) as a path separator:

Note:

- Correct: C:/Temp
- Not correct: C:Temp

As an example, to keep log data for two days in a directory called temp:

```
[General]
logDebug=true
logExpire=48
logDir=C:/Temp
```

Once you restart the client, you will find the log file in the <dir> defined in `logDir`.

Note: You will find the configuration file in the following locations:

- Microsoft Windows systems: `%APPDATA%\Nextcloud\nextcloud.cfg`
- macOS systems: `$HOME/Library/Preferences/Nextcloud/nextcloud.cfg`
- Linux distributions: `$HOME/.config/Nextcloud/nextcloud.cfg`

Alternatively, you can start the client in the command line with parameters:

1. To save to a file, start the client using the `--logfile <file>` command, where `<file>` is the filename to which you want to save the file.
2. To save to a directory, start the client using the `--logdir <dir>` command, where `<dir>` is an existing directory.

When using the `--logdir` command, each sync run creates a new file. To limit the amount of data that accumulates over time, you can specify the `--logexpire <hours>` command. When combined with the `--logdir` command, the client automatically erases saved log data in the directory that is older than the specified number of hours.

As an example, to define a test where you keep log data for two days, you can issue the following command:

```
` nextcloud --logdir /tmp/nextcloud_logs --logexpire 48 `
```

10.5.6 Nextcloud server Log File

The Nextcloud server also maintains an Nextcloud specific log file. This log file must be enabled through the Nextcloud Administration page. On that page, you can adjust the log level. We recommend that when setting the log file level that you set it to a verbose level like `Debug` or `Info`.

You can view the server log file using the web interface or you can open it directly from the file system in the Nextcloud server data directory.

Todo: Need more information on this. How is the log file accessed? Need to explore procedural steps in access and in saving this file … similar to how the log file is managed for the client. Perhaps it is detailed in the Admin Guide and a link should be provided from here. I will look into that when I begin heavily editing the Admin Guide.

10.5.7 Webserver Log Files

It can be helpful to view your webserver's error log file to isolate any Nextcloud-related problems. For Apache on Linux, the error logs are typically located in the `/var/log/apache2` directory. Some helpful files include the following:

- `error_log` – Maintains errors associated with PHP code.
- `access_log` – Typically records all requests handled by the server; very useful as a debugging tool because the log line contains information specific to each request and its result.

You can find more information about Apache logging at `http://httpd.apache.org/docs/current/logs.html`.

10.6 Core Dumps

On macOS and Linux systems, and in the unlikely event the client software crashes, the client is able to write a core dump file. Obtaining a core dump file can assist Nextcloud Customer Support tremendously in the debugging process.

To enable the writing of core dump files, you must define the `OWNCLOUD_CORE_DUMP` environment variable on the system.

For example:

```
` OWNCLOUD_CORE_DUMP=1 nextcloud `
```

This command starts the client with core dumping enabled and saves the files in the current working directory.

Note: Core dump files can be fairly large. Before enabling core dumps on your system, ensure that you have enough disk space to accommodate these files. Also, due to their size, we strongly recommend that you properly compress any core dump files prior to sending them to Nextcloud Customer Support.

CHAPTER

ELEVEN

FAQ

11.1 How the “Edit locally” functionality works

This functionality depends on the desktop client ability to register the mime to handle the nc:// scheme. That is the handler used by the server to open a file locally. This will allow the desktop client to open a document with the local editor when you click on the option “Edit locally” in your Nextcloud instance.

Note: Without properly registering the mime, independent of the browser and distro being used, the desktop client will fail to open a document with the local editor when you click on the option “Edit locally” in your Nextcloud instance.

The browser will warn you of the failure: “Failed to launch ‘nc://…’ because the scheme does not have a registered handler.”

11.1.1 How to enable it

In order to do that, you need to install the desktop client with the MSI installer on Windows or use a third party software to integrate the AppImage in your system on Linux.

11.1.2 On Linux

We use AppImage due to its universal compatibility but to take full advantage of the desktop client features you will need a third part software to integrate the AppImage in your system: we have tested AppImageLauncher and alternatively there is Go AppImage.

11.1.3 On Windows

The MSI installer will alter your system registry to register the mime to handle the nc:// scheme.

Alternatively, you can manually register the mime to handle the nc:// scheme:

1. Save the following content to a .reg file: ``` Windows Registry Editor Version 5.00

[HKEY_CLASSES_ROOTncshellopencommand] @=””C:\Program Files\Nextcloud\nextcloud.exe" "%1"” ``` 2. Double click on the .reg file to import it into the registry.

See https://nextcloud.com/blog/nextcloud-office-release-solves-document-compatibility-overhauls-knowledge-management/ for more information.

11.2 Some Files Are Continuously Uploaded to the Server, Even When They Are Not Modified.

It is possible that another program is changing the modification date of the file. If the file is uses the `.eml` extension, Windows automatically and continually changes all files, unless you remove `\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PropertySystem\PropertyHandlers` from the windows registry. See http://petersteier.wordpress.com/2011/10/22/windows-indexer-changes-modification-dates-of-eml-files/ for more information.

11.3 Syncing Stops When Attempting To Sync Deeper Than 100 Subdirectories.

The sync client has been intentionally limited to sync no deeper than 100 sub-directories. The hard limit exists to guard against bugs with cycles like symbolic link loops. When a deeply nested directory is excluded from synchronization it will be listed with other ignored files and directories in the "Not synced" tab of the "Activity" pane.

11.4 There Was A Warning About Changes In Synchronized Folders Not Being Tracked Reliably.

On linux when the synchronized folder contains very many subfolders the operating system may not allow for enough inotify watches to monitor the changes in all of them.

In this case the client will not be able to immediately start the synchronization process when a file in one of the unmonitored folders changes. Instead, the client will show the warning and manually scan folders for changes in a regular interval (two hours by default).

This problem can be solved by setting the fs.inotify.max_user_watches sysctl to a higher value. This can usually be done either temporarily:

```
echo 524288 > /proc/sys/fs/inotify/max_user_watches
```

or permanently by adjusting `/etc/sysctl.conf`.

11.5 I Want To Move My Local Sync Folder

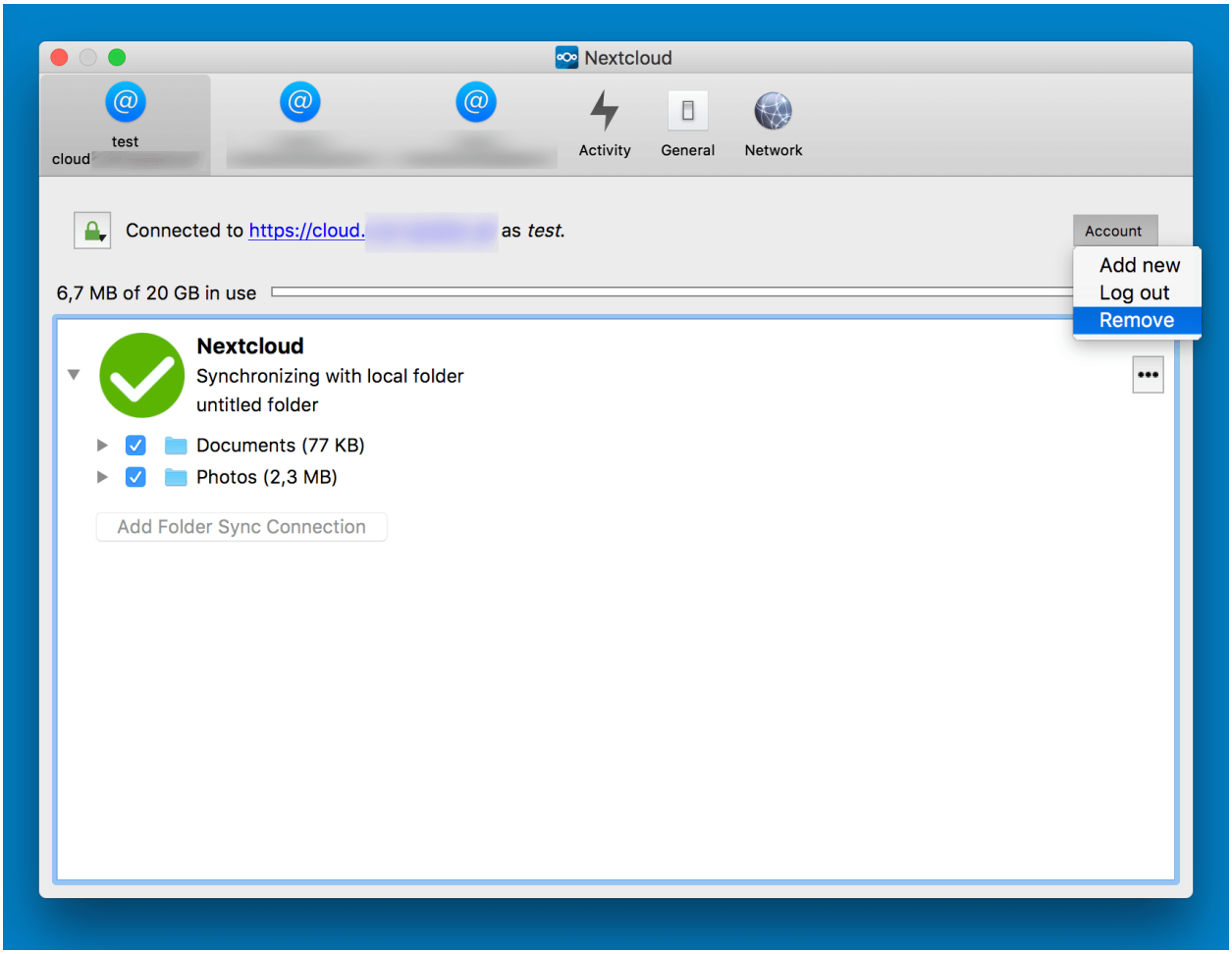
The Nextcloud desktop client does not provide a way to change the local sync directory. However, it can be done, though it is a bit unorthodox. Specifically, you have to:

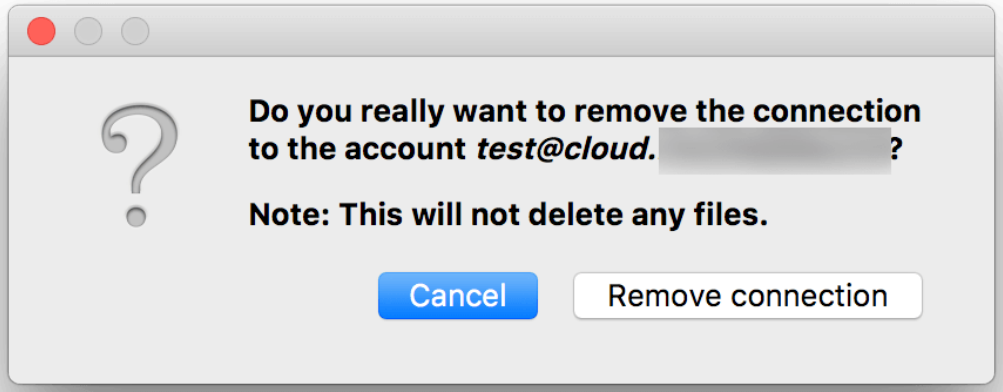
1. Remove the existing connection which syncs to the wrong directory
2. Add a new connection which syncs to the desired directory

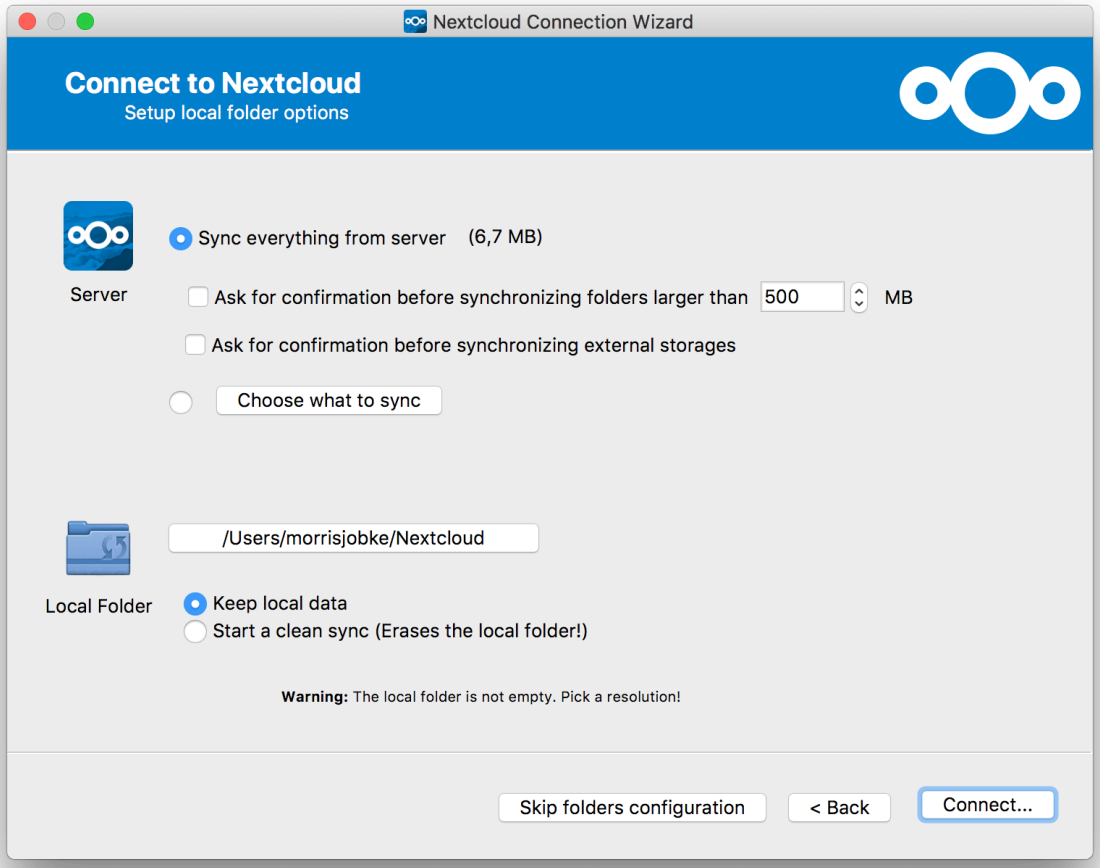
To do so, in the client UI, which you can see above, click the "**Account**" drop-down menu and then click "Remove". This will display a "**Confirm Account Removal**" dialog window.

If you're sure, click "**Remove connection**".

Then, click the Account drop-down menu again, and this time click "**Add new**".


Nextcloud
test
cloud
Activity
General
Network
Connected to https://cloud. as test.
Account
Add new
Log out
Remove
6,7 MB of 20 GB in use
Nextcloud
Synchronizing with local folder
untitled folder
Documents (77 KB)
Photos (2,3 MB)
Add Folder Sync Connection


Do you really want to remove the connection to the account test@cloud. ?
Note: This will not delete any files.
Cancel
Remove connection


Nextcloud Connection Wizard
Connect to Nextcloud
Setup local folder options
Server
Sync everything from server (6,7 MB)
Ask for confirmation before synchronizing folders larger than 500 MB
Ask for confirmation before synchronizing external storages
Choose what to sync
/Users/morrisjobke/Nextcloud
Local Folder
Keep local data
Start a clean sync (Erases the local folder!)
Warning: The local folder is not empty. Pick a resolution!
Skip folders configuration
< Back
Connect...

This opens the Nextcloud Connection Wizard, which you can see above, *but* with an extra option. This option provides the ability to either: keep the existing data (synced by the previous connection) or to start a clean sync (erasing the existing data).

Important: Be careful before choosing the "Start a clean sync" option. The old sync folder *may* contain a considerable amount of data, ranging into the gigabytes or terabytes. If it does, after the client creates the new connection, it will have to download **all** of that information again. Instead, first move or copy the old local sync folder, containing a copy of the existing files, to the new location. Then, when creating the new connection choose "*keep existing data*" instead. The Nextcloud client will check the files in the newly-added sync folder and find that they match what is on the server and not need to download anything.

Make your choice and click "**Connect…**". This will then step you through the Connection Wizard, just as you did when you setup the previous sync connection, but giving you the opportunity to choose a new sync directory.

CHAPTER

TWELVE

GLOSSARY

mtime
modification time
file modification time
: File property used to determine whether the servers' or the clients' file is more recent. Used only when no sync database exists and files already exist in the client directory.

Nextcloud Server
: The server counter part of Nextcloud Client as provided by the Nextcloud community.

Nextcloud Sync Client
Nextcloud Client
: Name of the official Nextcloud syncing client for desktop, which runs on Windows, macOS and Linux. It uses the CSync sync engine for synchronization with the Nextcloud server.

unique id
ETag
: ID assigned to every file and submitted via the HTTP `Etag`. Used to check if files on client and server have changed.

INDEX

V

W