Just a simple cloud

Expert’s manual

Revised: 2023-09-24

Contents

1. Just a simple cloud for Experts

1.1. Introduction

“**Just a simple cloud**” (JASC for short) offers you the possibility of keeping your **files** on **your own Raspberry Pi server** (RasPi for short) and **accessing** or **sharing** them **securely from anywhere in the world** with the aid of the client applications. At the same time, it makes sure that your files are always kept synchronised on all clients. The connection between the server and the clients is always fully encrypted so that only you have access to it. That means you keep **full control of your files** and do not have to make them available to a service provider with the risk of not knowing who else apart from yourself can access your files.

This manual supplements the Raspberry Server & Clients User Manual and explains the configuration of the system in detail.

2. Basics of operating a server

This section aims to provide an introduction to the basics required for operating a server on the Internet.

2.1. Addressing on the Internet

Every server that is accessible on the Internet is addressed by means of 2 parameters, which have to be known in order to access the server:

- the server address
- the server port

In most cases, the server address is written in a form that is easily comprehensible for us as humans; for example, our website address is:

www.just-a-simple-cloud.com

That format makes the address easy to remember. For a computer, however, the address has to be expressed as what we call an IP address. If we translate the above website address into an IP address, we get:

212.72.175.201

That conversion is performed by means of the **Domain Name Service** (**DNS**), which ensures that if a server is physically relocated from one data centre to another, the IP address is automatically updated to that of the new data centre. Fundamentally, the address can be used equally in either text or IP address format.

Since there are usually several services (e.g. a web server and an e-mail server) running on a server at the same time, a **port** number is also required to be able to distinguish which service is to be accessed. Most services use standard port numbers, e.g.

- Unencrypted web server (**http**://) Port 80
- Encrypted web server (**https**://) Port 443
- **POP3** e-mail Port 110

A more detailed description of the ports can be found at:

https://en.wikipedia.org/wiki/Port_(computer_networking)

2.2. Private Internet connections

The Internet connections of commercial operators and businesses are almost always assigned to fixed IP addresses. In the case of private connections, however, the Internet provider allocates a new (random) IP address every time an Internet connection is established. Even with connections that appear to be constantly connected, momentary disconnections occur followed by immediate reconnection and the issue of a new IP address.

If a server is operated on a private connection, therefore, the problem arises that the Internet address at which the server is accessible changes regularly and those changes are initially unknown.

2.3. Dynamic DNS

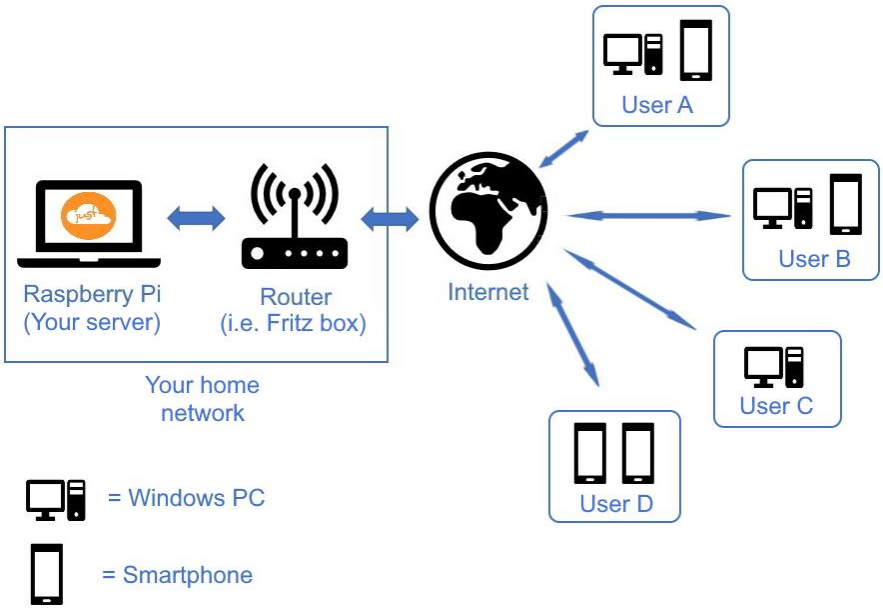
In order to be able to find the constantly changing IP addresses of a private connection, we use what is known as **dynamic DNS**.

The idea behind it is quite simple: a service provider keeps a database in which it links the unique name of the connection owner with the current IP address of the connection. To make that possible, a small program runs on the connection owner's computer and immediately notifies the database whenever the IP address of the connection changes.
So all that is required to find out the IP address of a private connection is to query the service provider's database and then you will always obtain the correct address.

With Just a simple cloud, that function is already integrated. As long as it is activated on the server, the IP address is automatically communicated to the clients.

2.4. Firewall

Virtually every Internet connection establishes a link between the Internet and the user's home network using a device called a **router**.



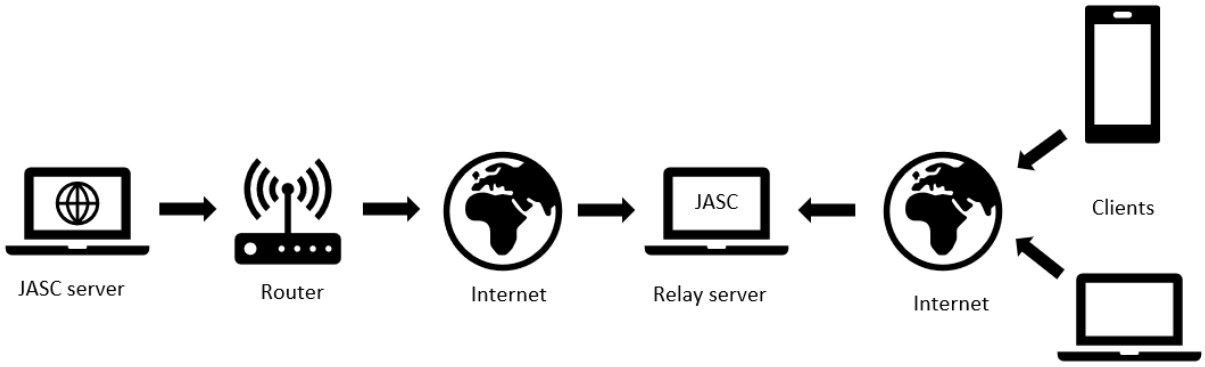
An essential task of the router is to provide a so-called **firewall**. The firewall protects the home network against unauthorised access from the Internet by blocking such access requests.

To operate a server "behind" a firewall, an appropriate exception rule ("**port forwarding**") has to be configured on the router. That rule defines how certain incoming connections from the Internet should be forwarded to a specific server within the home network.
Without such a rule, it is normally impossible to operate a server behind a firewall.

2.5. Relay server

The Just a simple cloud **relay server** makes both **dynamic DNS** and **configuration** of the **firewall** on the router superfluous.

To that end, the server (“**JASC server**”) establishes an **outgoing connection to our relay server** via the Internet.
On the relay server, the connection is **maintained** as long as a **client** is attempting to connect to **its JASC server**. The **relay server** then **relays** the data between the JASC server and the client.
That means that neither the server address of the JASC server nor its port number is required and port forwarding on the router can also be dispensed with.



And it goes without saying that the entire communication process is encrypted. Nobody can access the content of the data that is transferred via the relay server.

3. Configuration

Both the RasPi server and the Windows client are configured by means of an XML file (located in the sub-folder "**Config**").
The XML files are created by the Setup program and preconfigured with the most important settings. This section explains how you can manually optimise the configuration for your particular requirements.

3.1. Basic rules

The XML files are always subdivided into two sections:

- **General**: contains basic settings
- **Project**: contains settings that relate to the particular project concerned

An XML file can contain multiple projects. They are then numbered consecutively in the XML file. In other words, the first project will be contained within the section "Project 1", the next project in the section "Project 2", and so on. The projects must be numbered in ascending order without any gaps in the numbering.
The same applies by analogy to the section "User" within a project.

Example:

```
<?xml version="1.0" ?>
<JustASimpleCloudServer_Config>
  <General>
    <!-- License: enter the license key for the server here -->
    <License>inactive</License>
    <!-- Email Administrator: enter the email address of the Admin here, it will be used for notifications -->
    <Email>MyName@MyDomain.com</Email>
    <!-- GlobalKey: used to build up the session key. The global key will be internally combined -->
    <!-- with an internal secret key to exchange the session key at login. -->
    <!-- The GlobalKey must be identical on the server and all clients all over your installation -->
    <!-- The GlobalKey is encrypted, use PasswordGenerator to create a new password -->
    <!-- GlobalKey: initial key is "hello" -->
    <GlobalKey>37c533c8f4ce84808c19b94296e3cf06</GlobalKey>
    <!-- Remote Access: must be activated here: -->
    <ActivateRemote>true</ActivateRemote>
    <!-- Remote Access Passwort: initial password is "hello" -->
    <!-- "PasswordRemote" contains a hash of the password -->
    <PasswordRemote>454f431ca3a68536f90106ba4ddd3dde</PasswordRemote>
    <!-- "PasswordRemote_Encrypted" contains the encrypted password -->
    <PasswordRemote_Encrypted>37c533c8f4ce84808c19b94296e3cf06</PasswordRemote_Encrypted>
    <!- User privileges can be activated (= all folders are visible and writeable based on XML files) or de-activated (= default) ->
    <ActivateUserPrivileges>true</ActivateUserPrivileges>
    <!-- DNS service: if your server uses dynamic ip addresses, you can use our IP translation service: -->
    <ActivateDNSService>false</ActivateDNSService>
    <!-- Relay service: no need to configure port forwarding in your router, we will do that for you -->
    <ActivateRelayService>true</ActivateRelayService>
    <!-- Auto update: if actived, JASC will check once a day for updates and automatically install them -->
    <AutoUpdate>true</AutoUpdate>
    <!-- Enable webserver: the Apache webserver will be running anyway, but this flag controls, if the JASC server is connected
to Apache -->
    <EnableWebserver>true</EnableWebserver>
    <!-- shall the server automatically delete folders, that are empty (e.g. because the last file has been deleted from it)? -->
            <DeleteEmptyFolders>true</DeleteEmptyFolders>
    <!-- Where shall the server store a backup (if included in license)? -->
            <BackupFolder>./Backup</BackupFolder>
    <!-- Possible values for LogLevel: -->
    <!-- Error: critical errors preventing normal operation -->
    <!-- Warning: errors, which could have been repaired -->
    <!-- Info: all other informations -->
    <!-- IMPORTANT: The loglevel will not be updated, when you change it in the log dialog! -->
    <LogLevel>Error</LogLevel>
    <!-- LogUserActivity: indicates, if user log in/out, update of files shall be logged or not -->
    <LogUserActivity>true</LogUserActivity>
    <ScanIntervallUserrights>30</ScanIntervallUserrights>
    <TCPIPListenPort>50000</TCPIPListenPort>
  </General>
```

```
<Project1>
  <!-- Note: passwords are saved as salted hashes -->
  <Projectname>Testproject</Projectname>
  <Rootpath>ServerRoot/FirstProject</Rootpath>
  <Group1>
    <Groupname>Standard</Groupname>
  </Group1>
  <User1>
    <Username>Username</Username>
    <Realname>My Full Name</Realname>
    <!-- User passwort: initial password is "hello" -->
    <Password>37c533c8f4ce84808c19b94296e3cf06</Password>
    <Useremail>MyName@MyDomain.com</Useremail>
    <Group1>Standard</Group1>
    <IsAdmin>true</IsAdmin>
  </User1>
</Project1>
</JustASimpleCloudServer_Config>
```

Changes to the XML files are only applied once the server/client is restarted.

3.2. Configuration of RasPi server

3.2.1. General section

The section “General” contains the basic settings for the server:

- **License**: contains the licence code for the server; this is entered automatically. If the licence is updated, the changes are also applied automatically.
- **Email**: contains the e-mail address of the server administrator
- **GlobalKey**: contains a password from which the encryption for every single connection between the server and client is derived. Using that password, a “session key” is negotiated for every connection and used to encrypt communication for the session (= connection between server and client) until the connection it is terminated. The session key is generated using random numbers so as to ensure that the connection is secure.
- **ActivateRemote**: allows access using the tool “ServerRemote” (see relevant heading)
- **PasswordRemote**: the password (in the form of a hash code) for the connection with the ServerRemote tool
- **PasswordRemote_Encrypted**: the password (AES encrypted) for the connection with the ServerRemote tool
- **ActivateUserPrivileges**: (de)activates the handling of user permissions (see relevant heading)
- **ActivateDNSService**: (de)activates the DNS service (see relevant heading)
- **ActivateRelayService**: (de)activates the relay service (see relevant heading)
- **AutoUpdate**: (de)activates automatic updates. When this function is activated, the server checks once a day (at random times) whether an update is available and, if there is, installs it automatically
- **EnableWebserver**: (de)activates server output and user interface using the integrated web server
- **DeleteEmptyFolders**: (de)activates automatic deletion of empty folders on the server
- **BackupFolder**: specifies the path for server backups
- **LogLevel**: specifies the priority level of logs generated
- **LogUserActivity**: (de)activates generation of user activity logs
- **ScanIntervallUserrights**: interval in minutes at which user permissions are checked in the data directory
- **TCPIPListenPort**: even if the relay service is not used, the server is always accessible via the port specified by this parameter.

3.2.2. Projects section

It is possible (depending on the server licence) to create multiple projects at the same time. Each project then has its own storage area on the RasPi's SD card and is completely separate from the other projects. So, for example, one project might contain business files and another your personal photos without allowing access from one project to another.

A project is described by means of the following parameters:

- **Projectname**: unique name of the project; this name is used to identify the project on the clients
- **EnableHomeFolder**: activates automatic creation of a personal folder for each user in the "home" folder, see chap. **Home folder**
- **Rootpath**: path of the root directory containing all the project files
- **Group**: a list of user groups, see the section "User permissions management"
- **IsAdmin**: specifies whether the user concerned is an administrator
- **User**: parameters of an individual user:
  - **Username**: user name that is used to log onto the server
  - **Realname**: actual name of the user (optional)
  - **Password**: password of the user (passwords are stored on the server as "salted" hash codes)
  - **Useremail**: e-mail address of the user (optional)
  - **Group**: a list of user groups to which the user belongs, see the section "User permissions management"

3.2.3. Server backups

Provided the backup function is activated on the basis of the server licence, the server creates a backup of all data every day at midnight. The backup consists of the entire contents of "Rootpath" (see above) packed into a ZIP file.
The essential requirement for creation of a backup is that there is sufficient memory space available (usually on the SD card used).
The backup is placed in a separate ZIP archive for each day of the week, which is indicated in the name of the file. In other words, the backups are overwritten after one week at the latest.

The path to your backups can be configured in the XML configuration, section **General**: **BackupFolder**.

In case, there is not enough storage available on your SD card, the backup will be skipped.

3.3. Using external hard disk

In the default configuration, all files will be stored on the sd card of your RasPi.

In case, you need **more storage space**, you can use an external USB hard disk.

First, **stop the server on the web site**. After that, you can **transfer your files to the external hard disk**. You need to adjust the **server XML configuration file**: (see chap. 3) change the value **Rootpath** in the **project section**. As the last step**, restart** the RasPi.

Of course, you can also use **NAS devices** etc.

3.4. FTP Server

The RasPi already has a pre-configured FTP server:

- server name = JASC-RasPi
- port: 21
- user name: pi
- password: raspberry

3.5. Configuration of Windows client

You will find the configuration file of the client in this folder:

%LOCALAPPDATA%\JASC_Client

Example:

```
<?xml version="1.0" ?>
<!-- XML validator: https://www.w3schools.com/xml/xml_validator.asp -->
<JustASimpleCloudClient_Config>
  <General>
    <!-- ClientName: identifies this client on the server. Will be generated automatically during installation. -->
    <ClientName>8A473090B8B928B13EF9B1AF833D53C2C5104F46D39B871E4AF2E484AF890FF6</ClientName>
    <!-- GlobalKey: used to build up the session key. The global key will be internally combined -->
    <!-- with an internal secret key to exchange the session key at login. -->
    <!-- The GlobalKey must be identical on the server and all clients all over your installation -->
    <!-- The GlobalKey is encrypted, use PasswordGenerator to create a new key -->
    <GlobalKey>will be inserted here</GlobalKey>
    <!-- Relay service: no need to configure port forwarding in your router, we will do that for you -->
    <ActivateRelayService>true</ActivateRelayService>
    <!-- Relay service: enter machine idenifier of server here -->
    <RelayServiceIdentifier>will be inserted here</RelayServiceIdentifier>
    <!-- The client will show ballon messages on important notifications, if allowed -->
    <ShowBallonMessages>true</ShowBallonMessages>
    <!-- The client will try to set the root folder icon to the JASC icon, if allowed -->
    <SetClientFolderIcon>true</SetClientFolderIcon>
    <!-- Disc space warning level: a warning will be raised once, when the available disc space on the RasPi will go below this threshold -->
    <DiscSpaceWarningLevel>40</DiscSpaceWarningLevel>
    <!-- Disc space error level: an error will be raised once, when the available disc space on the RasPi will go below this threshold -->
    <DiscSpaceErrorLevel>20</DiscSpaceErrorLevel>
    <!-- Possible values for LogLevel: -->
    <!-- Error: critical errors preventing normal operation -->
    <!-- Warning: errors, which could have been repaired -->
    <!-- Info: all other informations -->
    <!-- IMPORTANT: The loglevel will not be updated, when you change it in the log dialog! -->
    <LogLevel>Error</LogLevel>
    <ScanIntervallProjects>1</ScanIntervallProjects>
    <ServerListenPort>0</ServerListenPort>
    <ServerAddress>unused</ServerAddress>
  </General>
  <Project1>
    <Projectname>Test</Projectname>
    <Rootpath>/home/stefan/JASC/rootfolder/</Rootpath>
    <Username>Stefan</Username>
    <Userpassword>will be inserted here</Userpassword>
    <!-- Folder filters: add the complete name of folders to be filtered out -->
    <!-- This filter is just an example, change the settings to your needs! -->
    <FilterFolder1>Temp</FilterFolder1>
    <!-- File filters: add a string, to be searched for -->
    <!-- This filter is just an example, change the settings to your needs! -->
    <FilterFile1>.tmp</FilterFile1>
    <LinuxUsername>stefan</LinuxUsername>
  </Project1>
</JustASimpleCloudClient_Config>
```

3.5.1. General section

The section “General” contains the basic settings for communication with the server:

- **ClientName**: name of the client; this is used for generating logs on the server and is subject to the maximum number of clients supported by the server on the basis of its licence
- **GlobalKey**: password for establishing an encrypted connection with the server (see relevant section)
- **ActivateRelayService**: determines whether a connection to the server should be established via the relay service or a direct connection (based on “**ServerAddress**” and “**ServerListenPort**”)
- **RelayServiceIdentifier**: if the relay service is to be used, the machine ID of the server must be specified here
- **ShowBallonMessages**: (de)activates output of important messages via pop-ups near the Windows system tray
- **SetClientFolderIcon**: if this setting is activated, the root directory of every project is marked by the Just a simple cloud icon in Windows Explorer
- **DiscSpaceWarningLevel**: a warning is issued if the remaining available space on the RasPi SD card falls below the percentage specified here
- **DiscSpaceErrorLevel**: an error is generated if the remaining available space on the RasPi SD card falls below the percentage specified here
- **LogLevel**: specifies the priority level of logs generated
- **ScanIntervallProjects**: specifies the interval in minutes at which the data directory (“Rootpath”) is checked for changes and, if necessary, a synchronisation process started
- **ServerListenPort**: server port for direct connection without using the relay service
- **ServerAddress**: server address for direct connection without using the relay service

3.5.2. Projects section

It is also possible to configure multiple projects on the client at the same time:

- **Projectname**: unique name of the project; this name is used to identify the project on the server
- **Rootpath**: path of the root directory containing all the project files
- **Username**: user name that is used to log onto the server
- **Userpassword**: Password of the user (passwords are stored on the client using AES encryption)
- **FilterFolder**: a list of directories that are ignored when synchronising with the server
- **FilterFile**: a list of file names or file name patterns that are ignored when synchronising with the server

4. Android app

This section describes the background to the Android app.

4.1. Battery optimisation

The Android operating system manages the life cycles of all apps in the background. The life cycle refers to the various statuses that an app can adopt; for example, it might be fully terminated then launched and displayed in the foreground. After that it might be moved into the background (e.g. by an incoming phone call) where it might stay until fully terminated again.

If the Android system resources (e.g. the RAM) are running short, the operating system decides which apps should be terminated. The apps themselves initially have no influence over that decision. That means that the JASC app might also be terminated by the operating system in certain circumstances.
To allow synchronisation to continue running in the background, the JASC app must be placed on the so-called “white list”. Apps on that list are allowed to continue running when resources are limited. This requires the permission “**REQUEST_IGNORE_BATTERY_OPTIMIZATIONS**”, which is automatically requested when the app is first launched. The app then tries to enter itself on the white list. Since, however, smartphone manufacturers often modify the operating system, there is no absolutely fool-proof method of doing so. Therefore, after installation you should check whether the app is actually registered on the white list.

Open the Android settings and navigate to “Battery -> App launch”. “Just a simple cloud” should be set to “Manage manually” on that list.

4.2. Permissions

Every Android app applies for various permissions from the operating system so as to be able to access critical functions. As a basic principle, those permissions should be kept as restrictive as possible. So the JASC app should not be allowed to access your contacts, for example. The following permissions are required for the app to function correctly:

- **CAMERA**: for scanning the QR code
- **INTERNET**: for accessing the RasPi server via the Internet
- **READ_EXTERNAL_STORAGE**: for accessing memory to read the project directory
- **RECEIVE_BOOT_COMPLETED**: so app can automatically be launched in the background if the phone is restarted
- **REQUEST_IGNORE_BATTERY_OPTIMIZATIONS**: see above
- **WRITE_EXTERNAL_STORAGE**: for accessing memory to write to the project directory

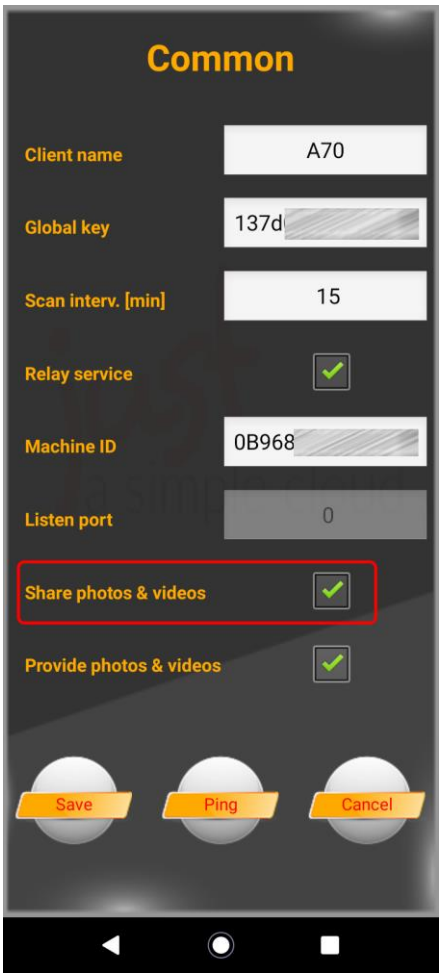
4.3. Share your pictures / videos

In the default settings, the app will automatically import and synchronize all your pictures, you have taken with the camera of the smartphone.

The pictures will be stored in the following path within the project:

Smartphone/<user name>/<device dame>

You can enable / disable the automatic import from the **common settings**:



If you want to remove pictures, delete them directly in the **DCIM** folder of your smartphone.

Attention: depending on your license, pictures and videos will be automatically handled:

- Personal license: **pictures only**
- Professional license: **pictures and videos**

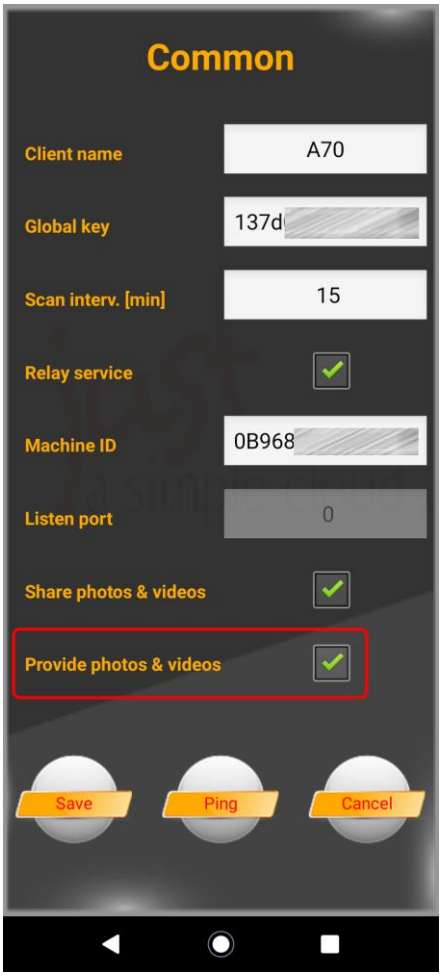
4.4. Provide pictures / videos

If there is more than one smartphone connected to your cloud, all pictures from the other smartphones, which have imported their pictures, will automatically be downloaded to your smartphone.

The pictures will be stored here:

Download/JASC

You can enable / disable this feature from the **common settings**:



This feature is **only available** on Android smartphones with **Android 11 or above**.

4.5. Android version 7 to 10

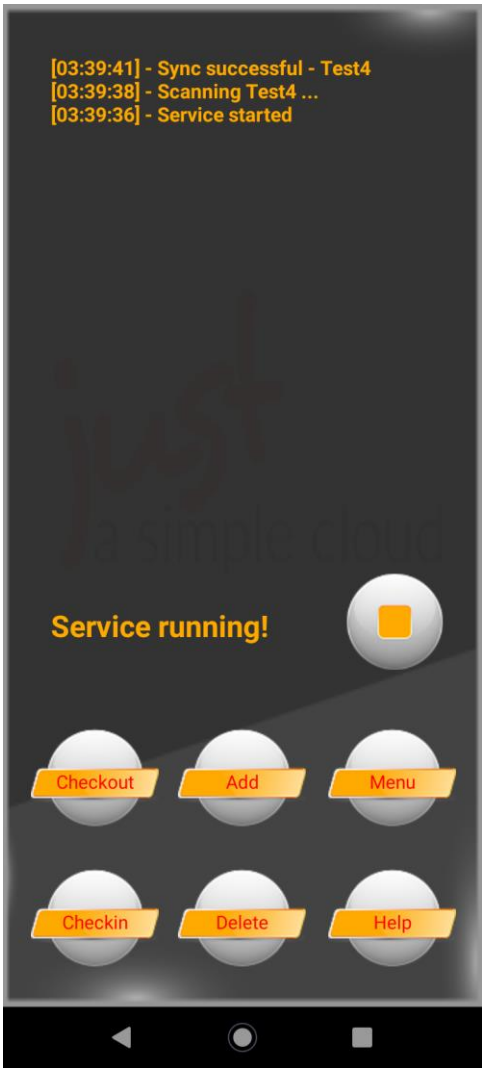
All Android versions up to 10 will allow access to files for all apps. This app will automatically create the following path and store all project files within this directory:

Internal storage/JustASimpleCloud

You will not find any additional functionalities on the main screen to access the files, use a file explorer instead.

4.6. Android 11 and later

Starting with Android 11, file access was restricted. It is no longer possible to create a common folder to store the files, providing access to all apps.
The app will instead store all files in an internal folder. In order to access your data, the app provides some additional features.



Checkout: will show up an explorer view of all your files. Click on a file to create a copy of it in the following folder:

Internal storage/Documents/JASC/<Projectname>

Now you can edit your file with every app on your smartphone. Files, which are checked out, will be highlighted in the view.
In case, a file is updated by **synchronisation** with the server, only the **master copy in the internal folder of the app will be updated**, your edited copied will remain unchanged.

Checkin: provides the same view as above. By clicking on a file, which was checked out before, it will be copied back to the internal storage of the app and automatically synchronised.

Add: use this function to add new files to your internal folder. As a first step, a file dialog will be shown to choose the file, afterwards, you can choose the folder, in which to store the file.

Delete: in case, you want to delete a file from the internal storage, you simple need to click on it in the view, which will be shown.

4.7. Misc

On smartphones from special manufactures, you will need to enable automatic launch here: Settings -> Battery -> Auto launch management -> App auto launch"

5. User permissions management

5.1. Requirements

Management of user permissions requires that the following conditions are satisfied:

- The parameter "**ActivateUserPrivileges**" must be activated in the server configuration.
- A meaningful value must be specified for "**ScanIntervallUserrights"** in the server configuration. The server will then periodically scan the entire data directory specified in the projects in the parameter "**Rootpath**".
- At least one user group name must be specified in the project parameter "**Group**".
- Every user assigned to a project must have at least one entry for "**Group**". That entry must match a group name specified in the project.

Important: if a project contains **no details** for "**Group**" and/or if **a user is not assigned to a group**, that means that the user concerned will have **neither Read Only nor Read/Write permissions for the entire data directory**!

Example:

```
<JustASimpleCloudServer_Config>
  <General>
    …. snipp …
    <ActivateUserPrivileges>true</ActivateUserPrivileges>
    <ScanIntervallUserrights>30</ScanIntervallUserrights>
    …. snipp …
  </General>
  <Project1>
    <Projectname>Testproject</Projectname>
    <Rootpath>ServerRoot/FirstProject</Rootpath>
    <Group1>
      <Groupname>Standard</Groupname>
    </Group1>
    <Group2>
      <Groupname>Admin</Groupname>
    </Group2>
    <Group3>
      <Groupname>Sale</Groupname>
    </Group3>
    <User1>
      <Username>John</Username>
      <Realname>John Doe</Realname>
      <Password>37c533c8f4ce84808c19b94296e3cf06</Password>
      <Useremail>john@doe.com</Useremail>
      <Group1>Standard</Group1>
      <Group2>Sale</Group2>
    </User1>
  </Project1>
</JustASimpleCloudServer_Config>
```

The following three groups have been defined for the project "Testproject":

- Standard
- Admin
- Sale

The user "John" is a member of the groups

- Standard and
- Sale

but not of the group "Admin". That means that directories reserved for the Admin group are not visible to John.

5.2. Access control

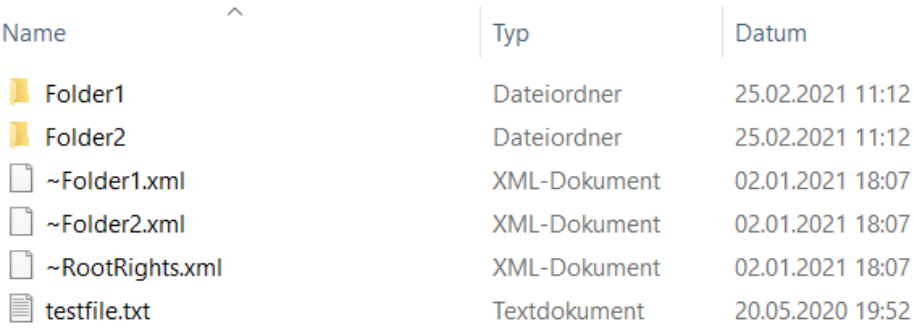
The access permissions are allocated at directory level. For each directory it is possible to specify for every group defined in the server configuration whether the group has Read Only or Read/Write access to the directory.

The permissions management settings are similarly stored in XML files with the following structure:

~Directory_name.xml

Example:

View of the **root directory** of a project:



For the two directories "**Folder1**" and "**Folder2**" there are the XML files "**~Folder1.xml**" and "**~Folder2.xml**" respectively.

The **access permissions for the root directory** of a project are always stored in the file "**~RootRights.xml**". That file must always exist.

If **no XML file yet exists** for a directory, it is **automatically created** the next time the periodical scan of the access permissions is performed. The access permissions are then copied over from the permissions for the immediately superior directory.

5.3. Structure of the XML files

Example:

```
<?xml version="1.0" ?>
<!-- this file contains the user rights for the folder -->
<UserRights>
	<Group1>
		<Groupname>Standard</Groupname>
			<Visible>true</Visible>
			<Write>false</Write>
	</Group1>
	<Group2>
		<Groupname>Sale</Groupname>
			<Visible>false</Visible>
			<Write>false</Write>
	</Group2>
	<Group3>
		<Groupname>Admin</Groupname>
			<Visible>true</Visible>
			<Write>true</Write>
	</Group3>
</UserRights>
```

Members of the group "**Standard**" have Read-Only permission for the directory (i.e. they can view files and subdirectories) but not Write permission.
Members of the group "**Sale**" have no permissions for the directory, i.e. they cannot even see the contents of the directory.
Members of the group "**Admin**" have full access permissions for the directory.

If a **group is not listed in the XML file**, that means that the members of that group do not have any access permissions for the directory (unless they have permissions by virtue of membership of another group specified in the XML file).

The group names do not have to be listed in the same order in the XML file as in the server configuration.

5.4. Home folder

In case, the property "**EnableHomeFolder**" is set to "true" in the project configuration of the server (see chap. **Projects section**), the server will **automatically** generate a **subfolder in the "home" directory** with the name of the subfolder identical to the username. This is valid for:

- RasPi server first time configuration
- Manage user on RasPi server

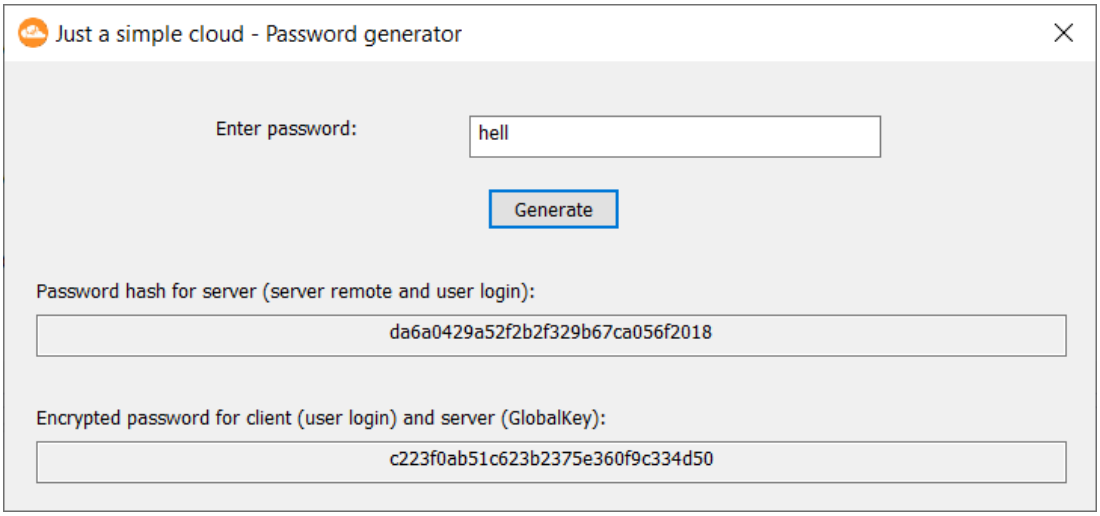
The server ensures, **only the user himself has read- and write access** to his personal folder and **no other user**. The rules for access control are not valid herein.

5.5. Administrators

Administrators (see "Manual JASC-RasPi", chap. "User management on the RasPi server") are **excluded from access control**.
Meaning, they always **have full read and write access in all folders**, independent from the rules of the access control or home folders.

6. Tools

6.1. Password Generator (Windows)



For security reasons, the user passwords are not stored in the XML configuration files as plain text. In the configuration file for the client, the password is stored using AES encryption; in the server file it is stored as a “salted” hash code.
If you want to create a new password (e.g. because the user has forgotten the old password), this tool helps you to generate the encrypted entries/hash codes. Simply enter the new password as plain text and then copy the value generated directly to the relevant configuration file.

7. Notes

Third party software

“VNC Viewer” and “balenaEtcher” are third party products and trademark rights of the respective owners.

7.1. Password Generator (Linux)

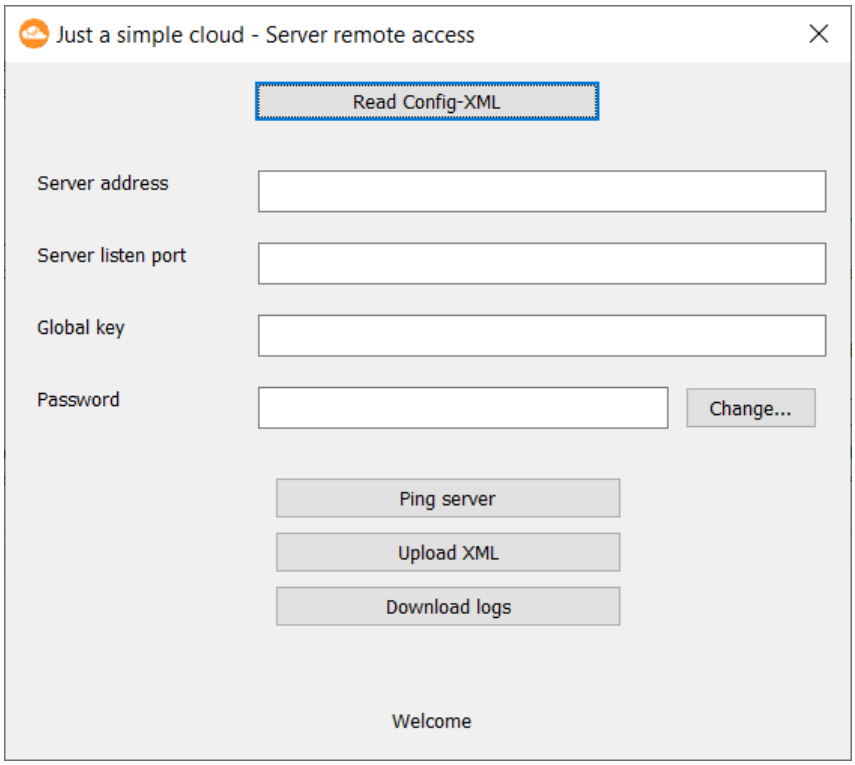
Use the JASC setup with the following command:

jascsetup password '<password>'

Always enclose your password in **single quotation marks**!

For security reasons, the user passwords are not stored in the XML configuration files as plain text. In the configuration file for the client, the password is stored using AES encryption; in the server file it is stored as a “salted” hash code.
If you want to create a new password (e.g. because the user has forgotten the old password), this tool helps you to generate the encrypted entries/hash codes. Simply enter the new password as plain text and then copy the value generated directly to the relevant configuration file.

7.2. Server Remote (Windows only)



To help you complete the server parameters, you can click the button “**Read Config XML**” and use the XML file generated when creating your users.

You can reset the password for remote access to the server by clicking the button “**Change...**”.

Communication with the server can be tested by clicking the button “**Ping Server**”.

This tool also enables you to update the configuration of a remote server. To do so, you create a new version of the configuration locally and upload the file “JustASimpleCloudServer.xml” directly to the server. The following settings in the **General** section of the configuration can be updated:

- Email
- ActivateUserPrivileges
- ActivateDNSService
- ActivateRelayService
- LogLevel
- LogUserActivity
- ScanIntervallUserrights

To update the user permission management files, you upload an XML file containing the changes. You do not have to upload every directory separately as all the directories listed in the XML file are automatically updated.

Example:

```
<UserRights>
	<Projectname>Testproject</Projectname>
		<Folder1>
			<Foldername>Rootfolder</Foldername>
			<Group1>
				<Groupname>Standard</Groupname>
				<Visible>true</Visible>
				<Write>true</Write>
			</Group1>
			<Group2>
				<Groupname>Sale</Groupname>
				<Visible>true</Visible>
				<Write>false</Write>
			</Group2>
		</Folder1>
		<Folder2>
			<Foldername>Folder1\Subfolder</Foldername>
			<Group1>
				<Groupname>Standard</Groupname>
				<Visible>false</Visible>
				<Write>false</Write>
			</Group1>
		</Folder2>
</UserRights>
```

After updating the settings, please wait for at least one server scan cycle to complete so that the new settings are applied.

To check your settings, you can **download the configuration file and all log files from the server**.