Etasoft 

Etasoft – XT Server 2.x

XT Server is a set of tools for automated data translation, validation and file processing.

Setup

Install software using setup program xtserver2.exe.
Package contains both GUI based and command line based processing tools.

First Steps

Plan your directory structure. All processing reads, moves and file writes. Having clear directory structure helps understand and diagnose problems.
One of the best ways to arrange directories is by creating main directory for specific job and then creating subdirectories below for individual tasks or groups of tasks.

Example:

1. You want to check directory for new files every 2 minutes.
2. Run validation on input EDI X12 835 files.
3. Separate failed and valid files.
4. Run translation on valid files to produce CSV files.
5. Send resulting output CSV files to remote FTP server.

Your first step is to create main directory.
Let say it is C:\prod\835tocsv .

Name contains transaction number and output format. If new transaction has to be added in the future it will be easy to do using same naming convention. For example: CSV to 837 translations can be placed into C:\prod\csvto837.
Exact naming details are not important. What is important is to be consistent and clear in naming.

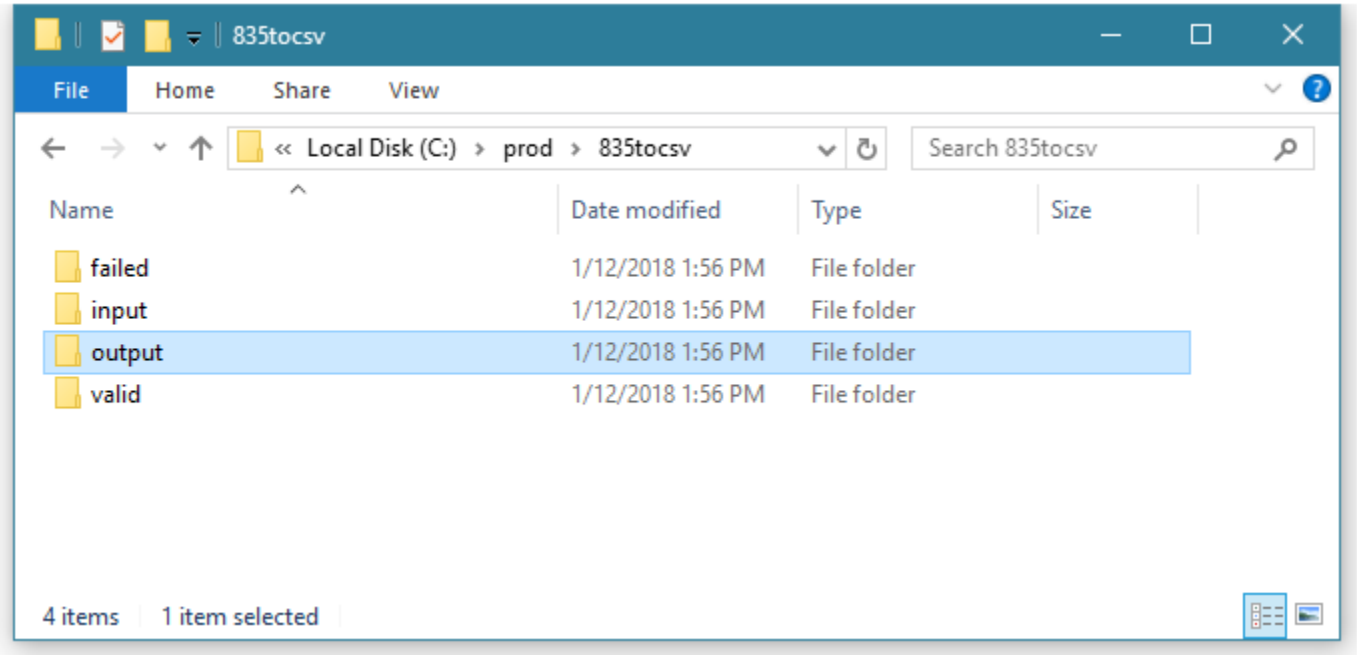
Once you create C:\prod\835tocsv add subdirectory called “input” to it. Your input files will be placed into C:\prod\835tocsv\input for processing.

There will be task to check for input files and then run validation. Validation task will split files into failed and successful. Create subdirectories for them.
C:\prod\835tocsv\failed
C:\prod\835tocsv\valid

After validation translator will run, take files from “valid” and put files into
C:\prod\835tocsv\output

Finally FTP task will send them out.
Resulting directory structure may look like this.

C:\prod\835tocsv\input
C:\prod\835tocsv\failed
C:\prod\835tocsv\valid
C:\prod\835tocsv\output

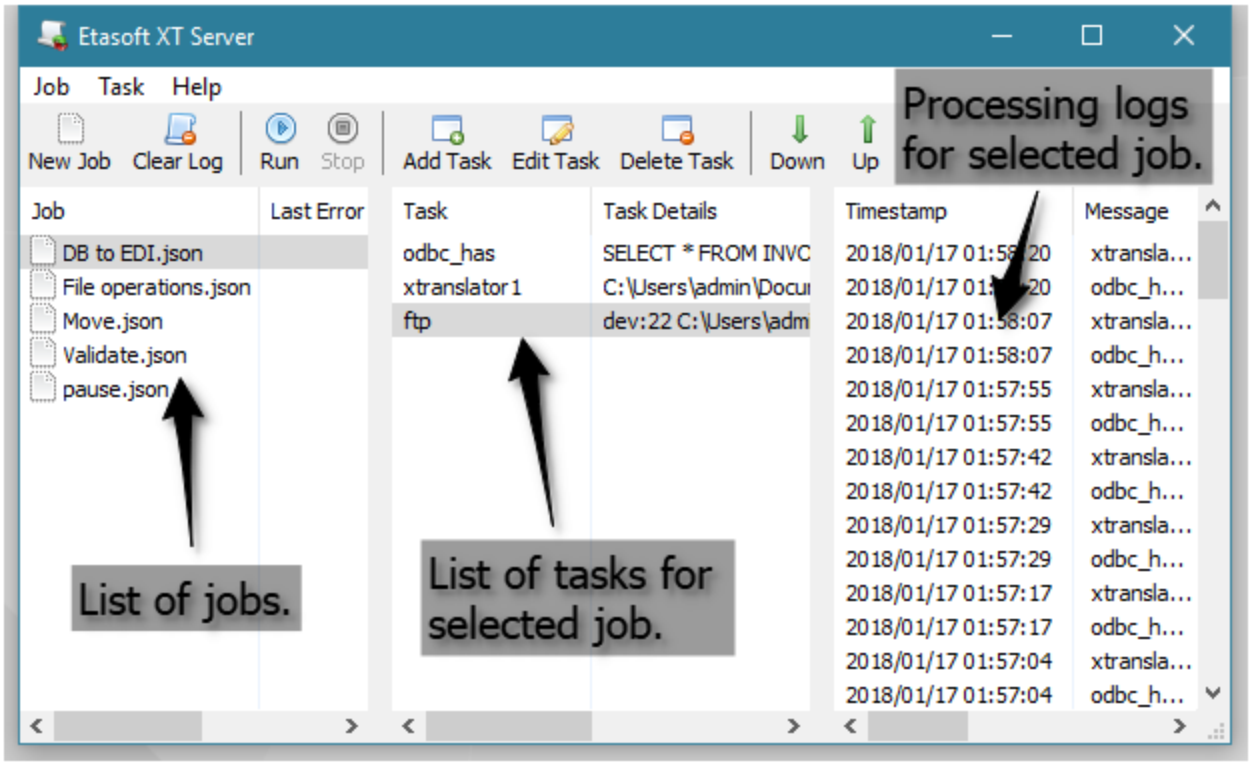


There is an example of final directory structure. Your setup may require additional subdirectories.

When starting new job avoid adding and setting all the tasks at once. Build it in small increments: add task, then test, add another task, test again. If you just added new task and job stopped working it should be easy to find the cause.

Job Tasks

XT Server runs jobs containing tasks. Each task is single unit of work such as file copy, translation, validation and so on.



Main window is divided into three panes: jobs, tasks and logs.

There are two fundamental types of tasks: actions and event watchers.

Event Watchers	**Actions**
File Polling	Email
File Watch	Execute Program
FTP	Exit
ODBC Has Data	File Copy
	File Delete
Schedule	File Move
Pause	
	FTP
	ODBC Execute
	Log
	Validator
	XTranslator 2
	Custom
	EDI X12 Split

1. Actions simply execute preset operation. Such as file move, validation or translation.
2. Event watchers wait for and execute other tasks below if some condition has been met. Watchers look for incoming files, specific database records or wait for scheduled time.

In rare cases task combines both action and event watcher. FTP is such task. It both looks for files and receives them. If there are no files to receive then tasks below are not executed and FTP task continues looking for files.

It is good idea to combine watcher tasks with time based tasks such as Schedule and Pause. Without Pause task FTP task will continue connecting to the server looking for files to receive without much pause. This might put strain on FTP

server running on the other side of the FTP connection.
Simply set Pause task to 120 seconds (that is 2 minutes) and have it just above FTP task.

Many FTP servers have special rules that prevent repeated polling without pauses. They use IP address based filtering. Once server notices multiple repeated connections from the same client it blocks all future connections from that IP address for extended period of time.

Jobs are meant to be short. If you have to run multiple translations using different types of files try to setup few jobs instead of using one very large job. Small jobs are easy to change and run. You can test each smaller job in isolation.

Each job should be setup as stand-alone unit of work. That way it can be tested independent of other jobs. Try to avoid creating dependencies between jobs: such as one job reading input files that output from other job. Inter-dependencies between jobs create hard to diagnose concurrent execution issues.

Typical setup may contain 4-10 different jobs. Each job may start with scheduled event or pause followed by event watcher followed by few action based tasks.

Typical job:

Schedule or Pause task	This is used to add some time between file or FTP or database polling for new data.
Event Watcher (File Watch, ODBC Has Data, FTP)	
Actions (File Copy, Xtranslator2, Execute Program)	
Exit task	This is used for testing and manual processing only.

If there is no Schedule or Pause task before (above) Event Watcher then most Watcher tasks will poll for new data every 10 seconds. One exception is File Watch. It does not poll but uses file system events and should trigger in less than a second.

Processing

Job runs in implicit loop. If job runs all the tasks or there is no event for watcher to act on job starts from task #1 again.

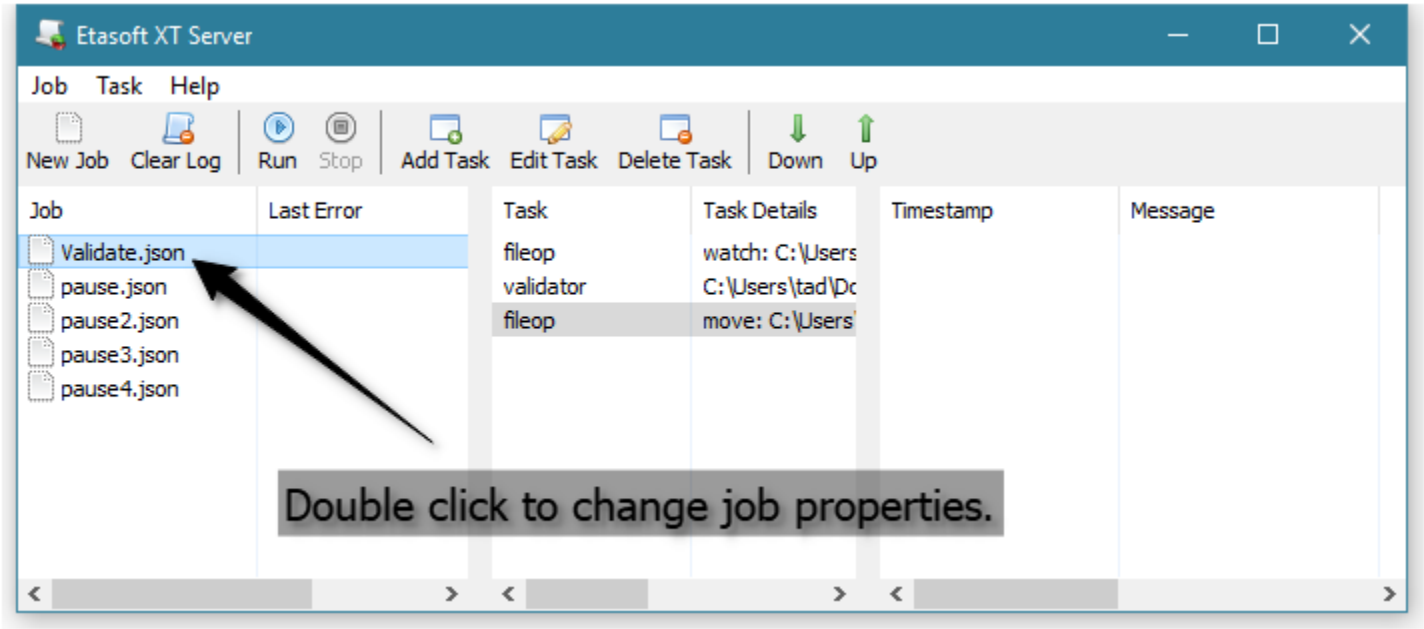
It is designed with idea that processing jobs meant to run in perpetual processing loop waiting for events. Once event gets triggered job runs action tasks and goes back to the beginning waiting for more events. If you want job to exit this loop simply place Exit task at the end of all tasks for that job.

During initial production deployment it makes sense to switch on verbose logging for the job.

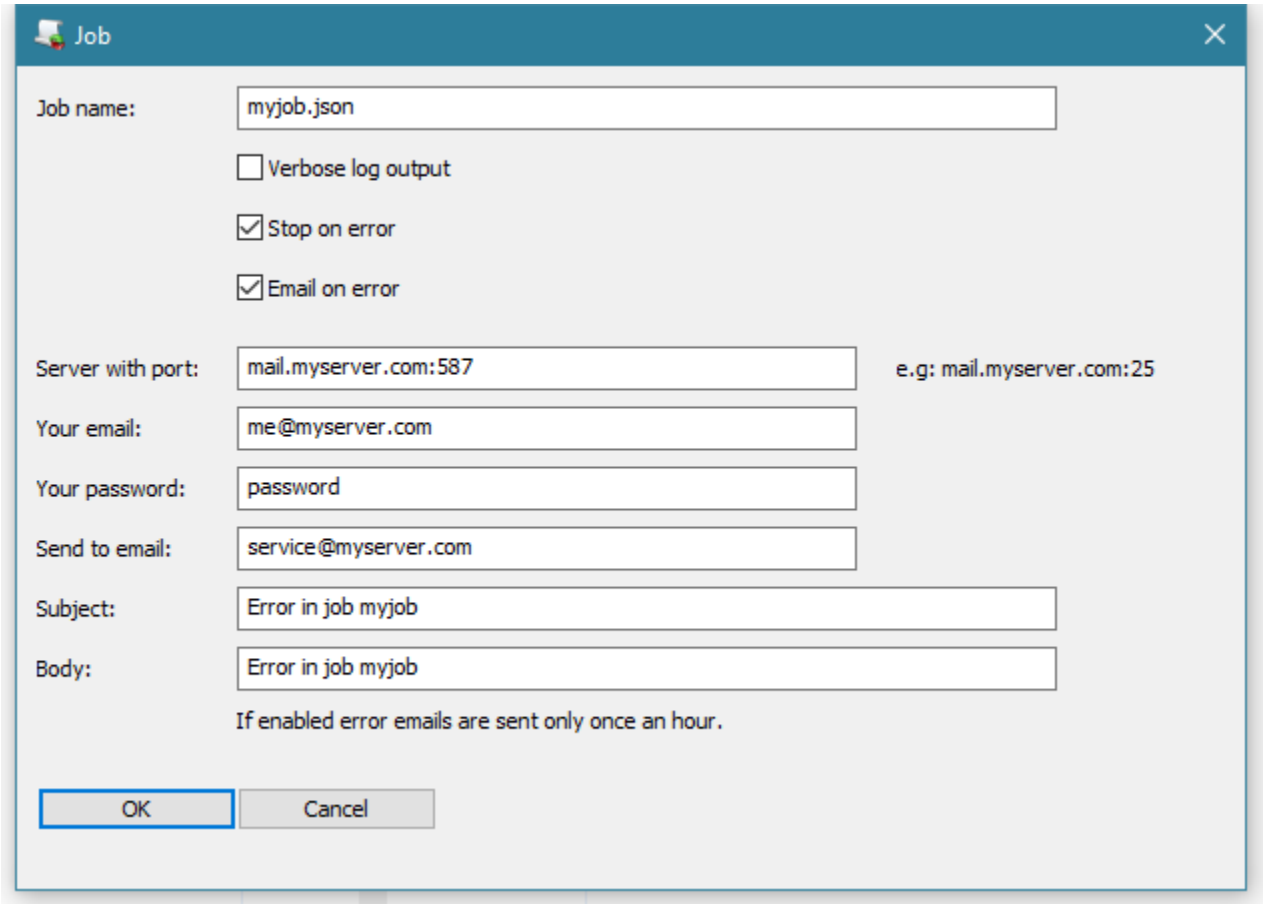
Implicit loop also means that job will continue running until it is stopped or error happens. Table below lists all scenarios when job exits loop.

Job stops and exits out of the perpetual loop if:
1. Exit task runs.
2. One of the tasks fails with error and “Stop on error” option is set in job properties screen.
3. One of the tasks fails with FATAL error. Fatal errors are rare. Please contact support if such error happens. Email us file with job name ending with “_fatal.trace”.
4. Job is stopped manually via GUI or Ctrl-C is pressed when running job using command line interface.

When any job is executing number of GUI add and edit operations are disabled. If you want to add or edit tasks stop all the jobs.



When none of the jobs are running double clicking on job or task opens job's properties screen.



"Stop on error" flag changes how errors are treated by executing job. If this flag is unchecked errors will be ignored and job will restart from task #1 once any error is encountered.

Many errors are network related. Such as sending or receiving files over the network and FTP. In such cases automatic job restart may be acceptable and un-checking “Stop on error” makes sense.

If you have a job that does not deal with network files stopping on error is recommended. Since errors in such cases might be critical and restarted job will keep on failing. For example: running out of disk space, having incorrect connection information for ODBC.

“Email on error” sends email notifications on job failures. In order to avoid too many error emails only first notification every hour is sent. When error happens it is sent without delay but all additional errors that same hour are not emailed. If you have intense processing then errors might accumulate every quickly resulting in hundreds of emails. One email per hour filter prevents mass emailing.

File Tasks

Use file tasks for basic file operations. If files are on the network accessible via mapped network drives or NFS there is a chance that operation may fail.

Both File Polling and File Watch check for incoming files. File Polling uses time interval to check for files and only works on main directory it is setup for. If main directory has subdirectories it will not react to files inside nested subdirectories.

File Watch uses combination of file system events and polling. File Watch is faster than File Polling task because file system events are almost instantaneous. Some network drives do not support file system events and File Watch will fail or will be slower to pick up files from them.

File Move task can be set to move limited number of files per each run. This sometimes required for translation and other tasks that have to limit output file sizes based on strict rules. This way File Move produces sequence of batches of files feeding them into tasks below.

It is idiomatic and correct to start your job with File Watch or File Polling and then add other tasks below.

When dealing with network drives and slow connections keep in mind possible data races. If you have slow process that copies files for you to access them your File Watch may react to incoming file faster while external process is still copying file for you to pick up. One simple way of solving it is to add Pause task right after Watch.
There are other techniques on how to solve this. Please contact product support if you encounter such problems.

Many tasks are designed to move files. It is no accident. Files have to “flow” through the system from one directory into the next. If files do not get moved then there is a risk that when job restarts or reenters loop again, it will pick up same files and reprocess them causing data duplication.

FTP Task

FTP connections are slower than local file access. Therefore FTP task combines both watch and action components.

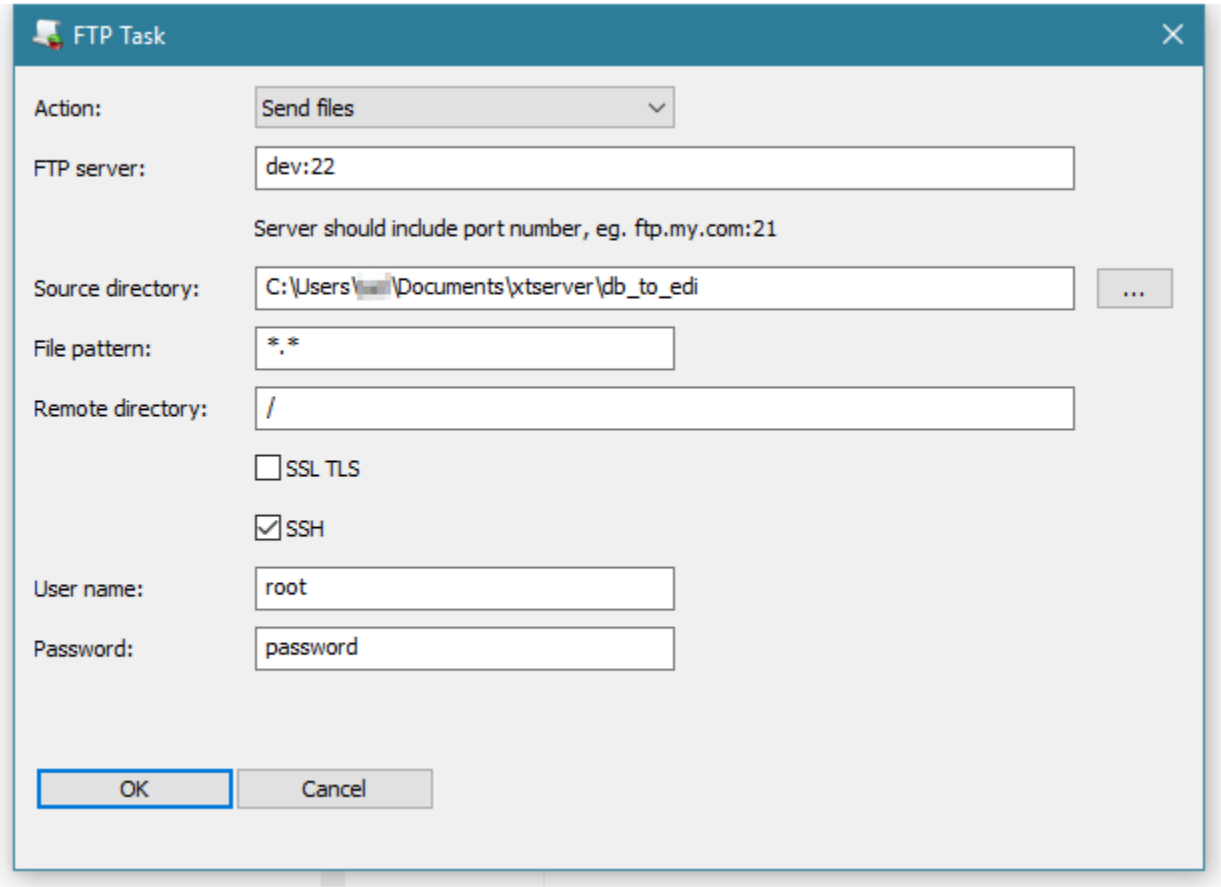
XT Server supports following FTP protocol extensions:

1. Simple FTP with plain username and password via port 21.

2. Secure FTP via SSH (SFTP). Typical secure connection is via port 22. It is mostly used for connecting to servers running Unix, Linux systems.
3. Secure FTP via SSL TLS (FTPS). Requires that FTP server should have SSL certificate installed. Certificates have expiration date. If certificate expires or is not valid FTP will fail.

Out of this list #2 is both easy to setup and secure. #1 trades security for ease of use. #3 seems to be least popular option since it requires skillful administrator. SSL certificates come in number of encoding formats making certificate installation more complex. Certificates also expire making it constant maintenance issue.

When setting up FTP task start with basic job containing only single FTP task. Test it. Make sure it can connect and send or receive files. Then expand on it.



It is important to have server address, port number and remote directory right. If you get server or port number incorrect then FTP task will fail fast and give you connection “dial” error.

Remote directory is usually difficult to get right. In most cases “/” or “.” works and indicates current directory on FTP server. You may need to attempt transmit test file few times using different value for Remote directory in order to see test file appear on the FTP server.

Typical plain FTP connection uses port 21. Simply keep both SSL TLS and SSH unchecked.
FTP task removes files from the source directory. This is done on purpose. Otherwise repeated task execution would fetch same files again causing duplicate data processing.

FTP SSL TLS

This is most challenging option out of all FTP configurations. X.509 certificate has to be installed on the FTP server to make it work. You can use self-signing certificates. Self-signed certificates contain private keys generated on local computer or on your server. You do not need to purchase them from official signing authority.
There are number of utilities that generate self-signed certificates. One of them is called “openssl” and is available in many Linux distributions.

Certificates are containers that hold private or public keys. Consult your FTP server documentation to find out what certificate format it needs. Typical formats are “pfx”, “cer”, “crt”, “p7b” and “pem”.

“pem” format is popular on Linux FTP servers while “pfx” and “cer” are typical Windows certificate formats.
“openssl” also converts certificates from one format into another. If you have certificate in one of the formats and your FTP server requires it in another format then simply convert it.

XT Server uses implicit SSL TLS mode and does not need certificate itself as long as FTP server has certificate installed and allows “implicit SSL TLS” connection.

Important: certificates expire. Usually in 1 year.
After 1 year most knowledge of FTP server setup is usually forgotten and trying to restore data transfer may take few frantic days. You may be able to post-pone this by installing certificates with expiration date way ahead in the future, e.g. 5 years. Even better option is to setup certificate auto-renew process.

FTP SSH

This option does not require X.509 certificates on the server. But not all FTP servers support SSH connections. If you have server running Linux/Unix OS it is usually not a problem.

Schedule and Pause Tasks

Schedule and Pause Tasks provide time based events. Schedule blocks until set time or time interval arrives, while Pause blocks only briefly. Both tasks might be combined with other watchers to get desired processing.
Schedule or Pause Tasks are usually very first tasks in the job. They trigger time based events for FTP, ODBC and other tasks.

ODBC Tasks

Use ODBC Execute task to update or delete temporary records. Use ODBC Has Data task to start database driven processing. ODBC Has Data task should be combined with Schedule task.
If SELECT statement in ODBC Has Data task returns non-zero result set then processing will continue running other tasks below it.

ODBC connection has to be setup as Data Source inside ODBC Control Panel. That Data Source then entered into ODBC Task as connection string in a form of DNS={data source};UID={username};PWD={password}.
Replace text together with {} using your specific connection values. Example:

DNS=MySQLServer;UID=bob;PWD=verysecurepassword

Validator Task

Validator task runs EDI validation on input or output files. Mostly input. It can separate failed and valid files into separate directories and produce output reports in number of formats such as CSV, HTML, X12 997 and X12 999.
X12 997, X12 999 and TA1 are EDI acknowledgements. They should be sent back to the trading partner sending you EDI files.

EDI validation is built-in and does not require separate license.

XTranslator 2.x Task

XTranslator2 task runs XTranslator version 1.x and 2.x maps. You should have valid Translator license to run maps.
Path to map file, input and output supplied to translator should not have any space characters in it. In general it is advisable to have no spaces in all processing paths when setting up jobs. Spaces make command line execution harder.

If 'Treat translation warnings as error' is checked then any warnings during translation will be treated as error inside XT Server. Typical warnings during translation get produced for unrecognized segments that do not match map's input side. This setting allows stricter processing to be turned on.

XTranslator Lite Task

XTranslator Lite task runs XTranslator Lite version 1.x maps. You should have valid Translator license to run maps.

Custom Task

Custom task runs external program and passes two parameters to it via -input ("dash input") and -output (" dash output") command line switches. This is used to hook up external custom tasks that are very specific, like filtering input files based on custom text inside them.

Custom task executes external command line program and reads its output. If program returns non-zero exit code or returns text "error" in standard output or standard error streams then job execution stops with error.
In other words: if you are building custom task and want job to stop execution then return non-zero exit code or write "error" into stdout or stderr streams.

Exec Task

Exec task runs external command line (batch) program. If program returns non-zero exit code then it is treated as error and job execution stops at that point.

EDI X12 Split Task

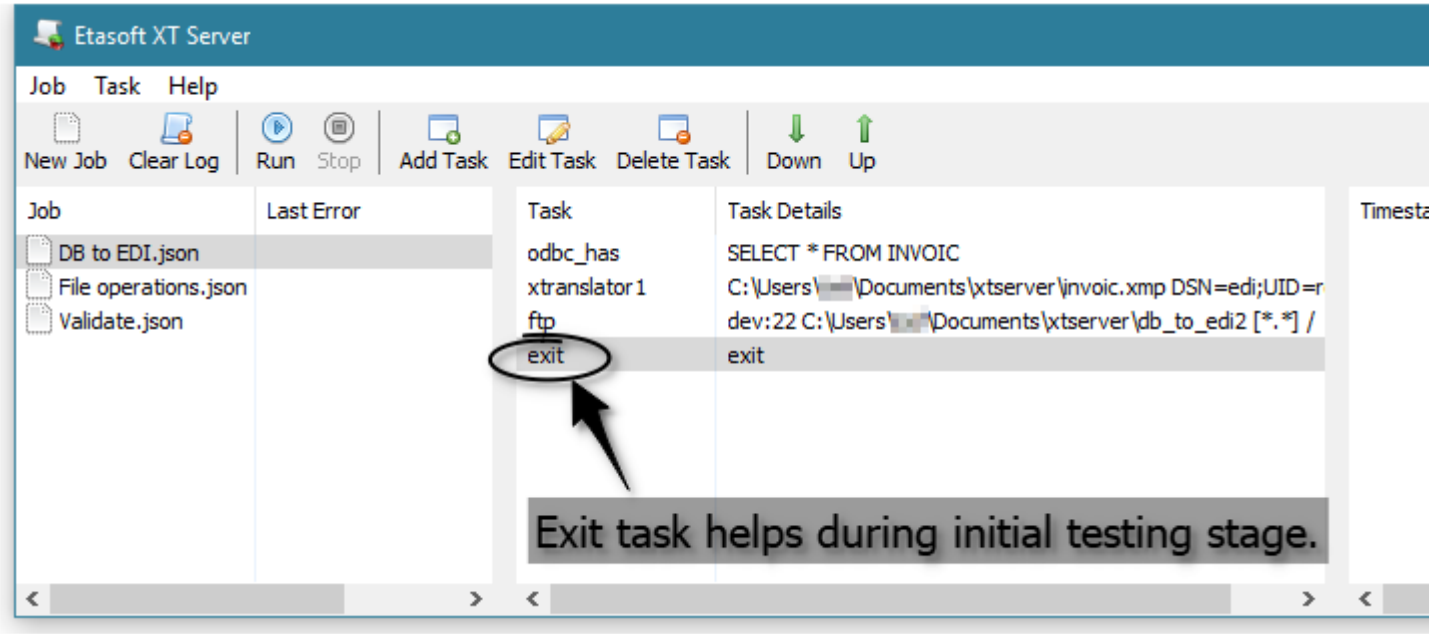
X12 Split task takes files that match wildcard name pattern inside input directory and splits them into separate files each with single transaction (single ST/SE). Output files get produced into output directory. This task primarily used for processing large EDI X12 files of 100Mbt or more.
Once large file gets split into smaller files it can be easier processed by translator or validator. Multiple small files get processed much faster than one very large file.

EDI X12 Split task cuts ST/SE out of the input file but also re-uses original X12 envelope of ISA/GS/GE/IEA segments that wrapped original ST/SE transaction.

X12 Split will not work on invalid EDI X12 files.

Exit Task

Exit task stops job execution and exits perpetual loop. This is helpful for semi-automated or manual jobs.



It is also helpful during testing when you want to run job only once and then check processing to make sure all files have been moved to correct destinations.

Miscellaneous Tasks

If you have issues setting up jobs use Log task to write extra information in the output logs. Set unique and meaningful text inside Log task.

Add Log tasks before and after task that is not working. Extra logging information may help diagnose the problem. There is also global “Verbose log output” option at the job level. Set it on if you are having issues with job execution.

Use Exec task to run additional external programs as part of job processing. Exec task reads external process exit code.

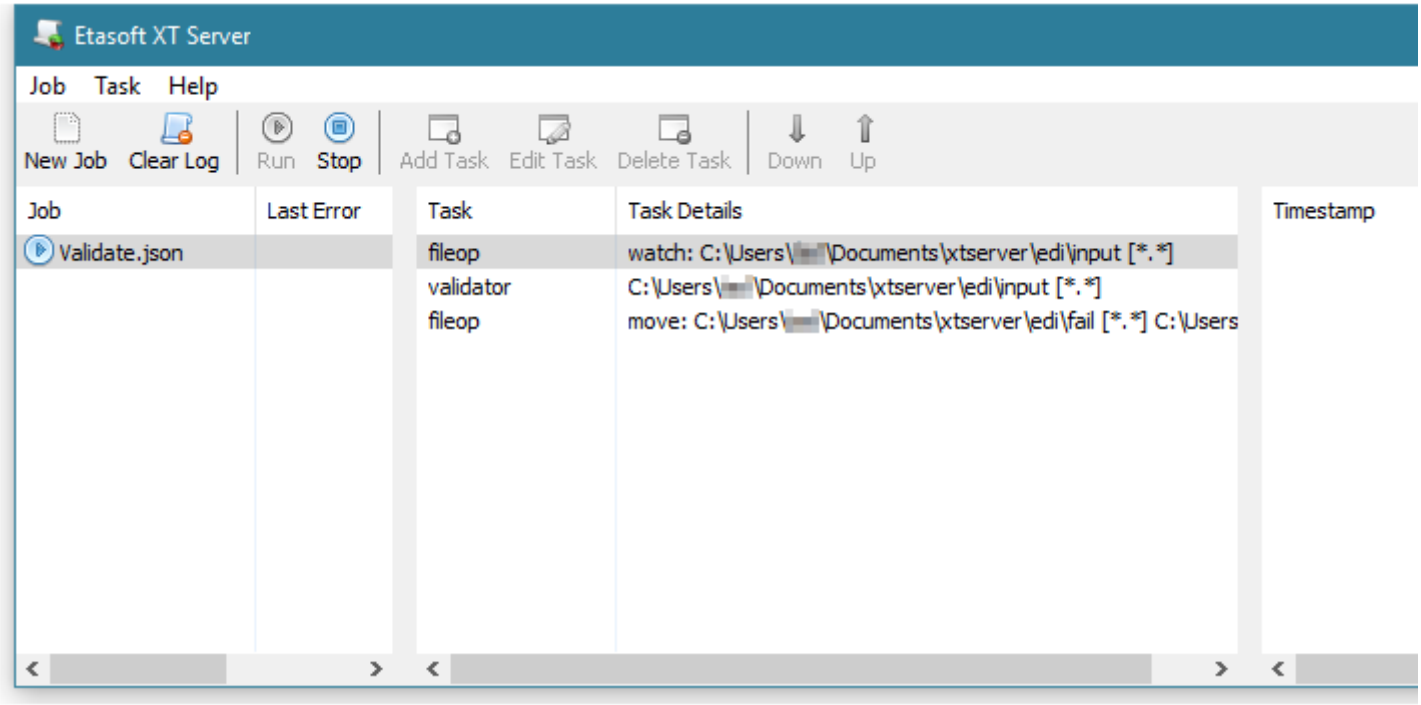
Email task is used for sending email notifications. Typical use is for over-night job monitoring.

Example Validation Job

This basic example performs validation and moves files.

Validation job starts with File Watch on directory C:\Users\admin\Documents\xtserver\edi\input.
File pattern is *.* so any file could be picked up.

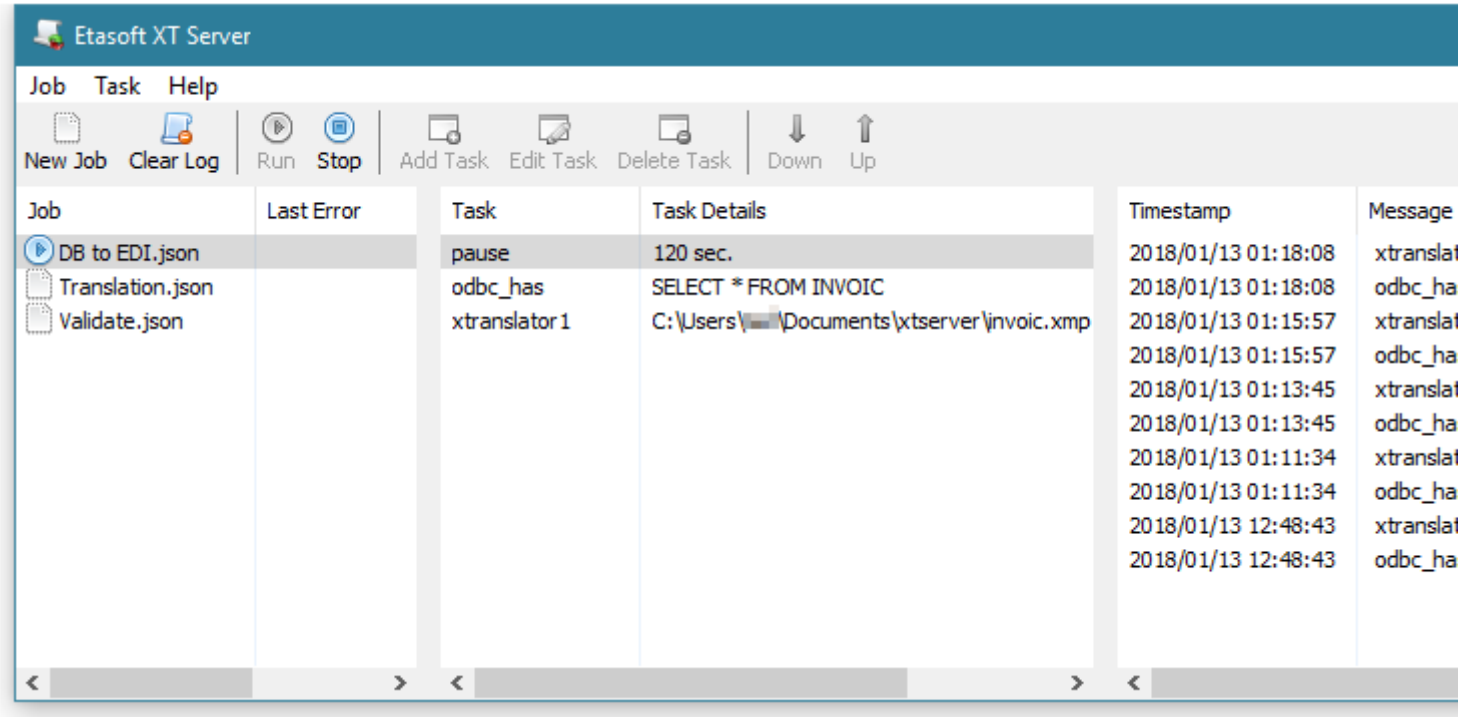
After File Watch triggers that there are files to process Validator task runs and separates files with failed files placed into C:\Users\admin\Documents\xtserver\edi\fail.
Last File Move task moves failed EDI files for administrator to look at them.



There is example of basic Validation job.

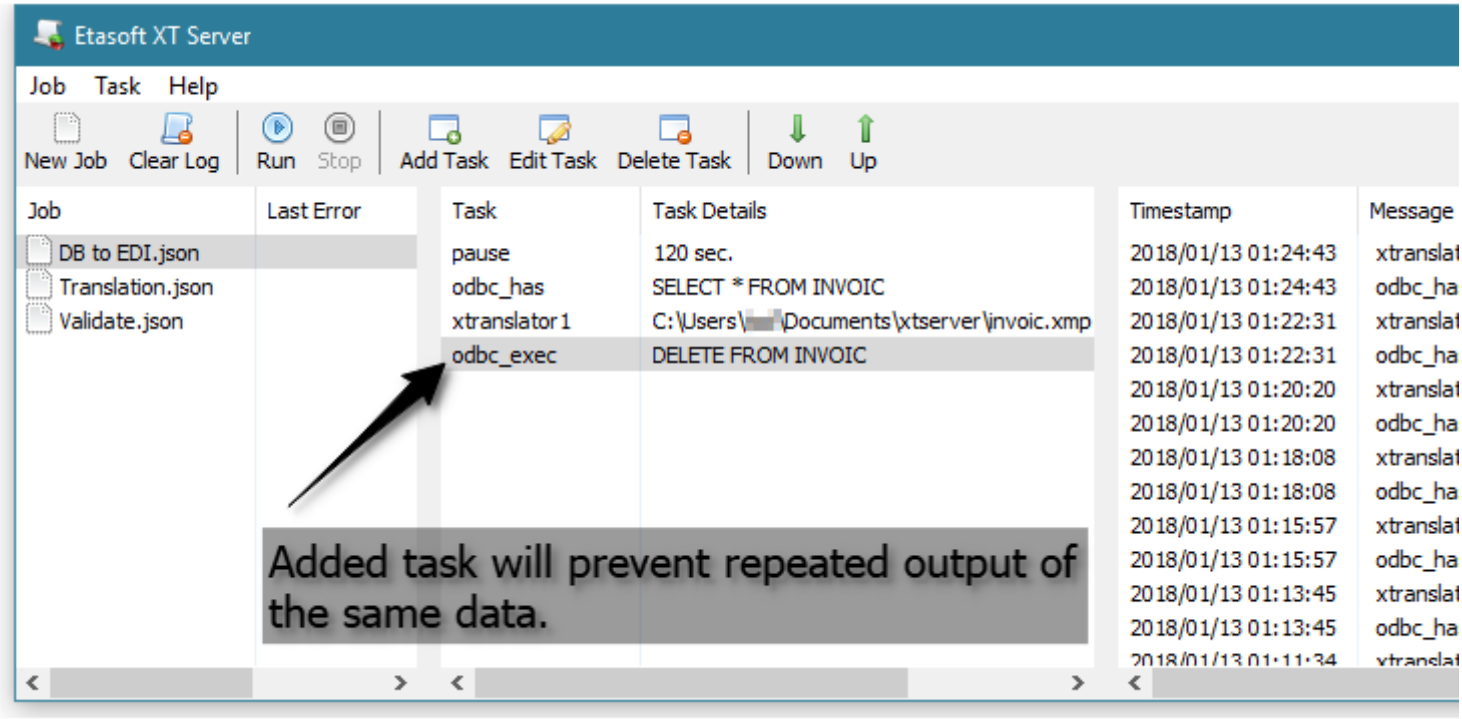
Example Translation Job

This basic example checks database for new records every 2 minutes. Once records get inserted into the database by external process translation runs and produces output file.



This example is not complete because after database records are detected it will keep on producing output file in the loop.

One way to fix infinite production of output files is to delete database records that have been processed. Add ODBC Execute task to the end of task list to delete records.



This fixes infinite output problem by deleting temporary processing records from the database.

ODBC Execute works here, but when you are dealing with files and do not want to process same input repeatedly use File Delete or File Move tasks to clean input directory. FTP and Validator tasks auto-clean after processing and do not need extra tasks to clean after them.

Translator map can be setup to delete input files via internal option inside translator Map Editor. Option is called “DeleteWhenDone”. If option is not setup Translator task will not delete input files and File Delete task would work.

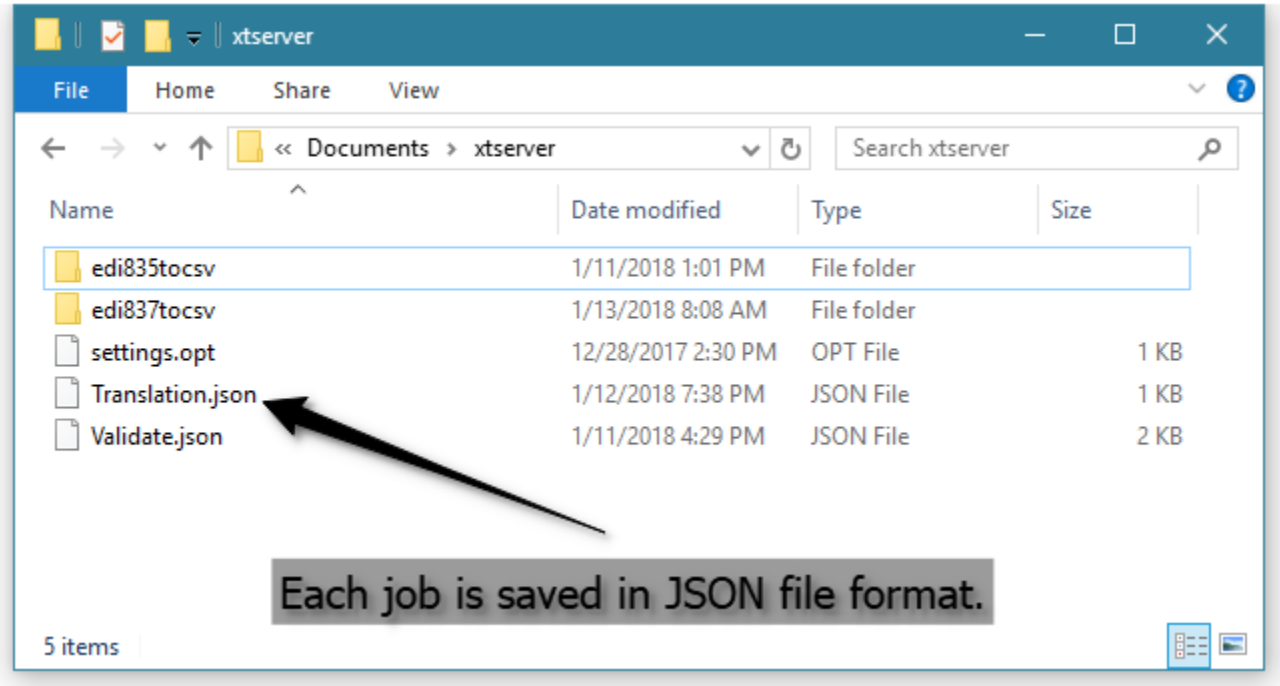
Job Files

Jobs configuration data is saved into JSON files. When moving to new hardware simply copy whole xtserver directory into the same location on new computer.

You can also use this to create many similar jobs. Create one typical job using GUI and close it. Then navigate to xtserver directory to find JSON files. Make copies of the JSON files into the same directory and rename them. Start GUI again. It will load all copied jobs with new names.

During processing each job saves its own log into CSV file. Example: if you have “edi_to_xml.json” job file then log for that job will be saved into “edi_to_xml.csv”. CSV files are easy to load into relational databases and analyze. Once log is

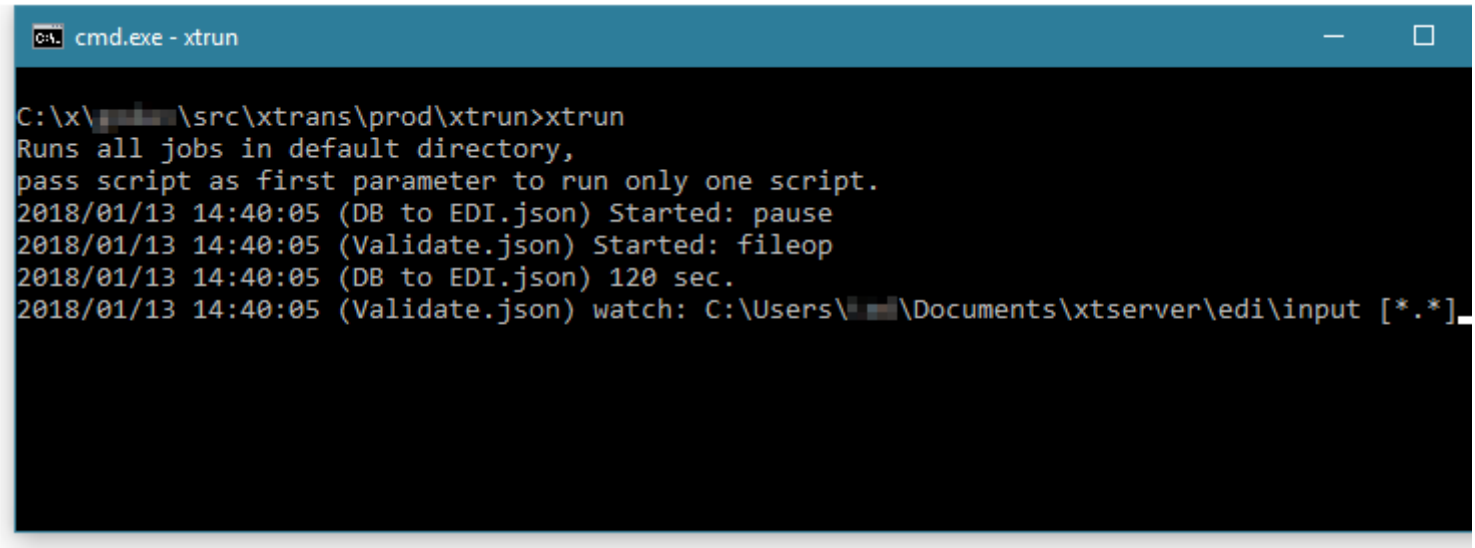
in the database you can filter records and have report listing only input and output files that have been processed by the job.



JSON files contain editable text. But it is not recommended to edit them using text editor. Format is sensitive and easy to break if you are not familiar with it.

Command Line Tool

There is command line tool to run server configuration setup via GUI. Tool is called “xtrun”. It runs all jobs by default if no parameters are passed into it.



Command line tool has few advantages. You can schedule it to start when computer starts using Schedule Tasks (part of Windows). It is also more direct, there is no need to point and click. When tool starts it will run all the jobs. All activity is logged on to a console terminal. If you want to have it saved in to a file simply use Windows OS output redirection when starting “xtrun”.

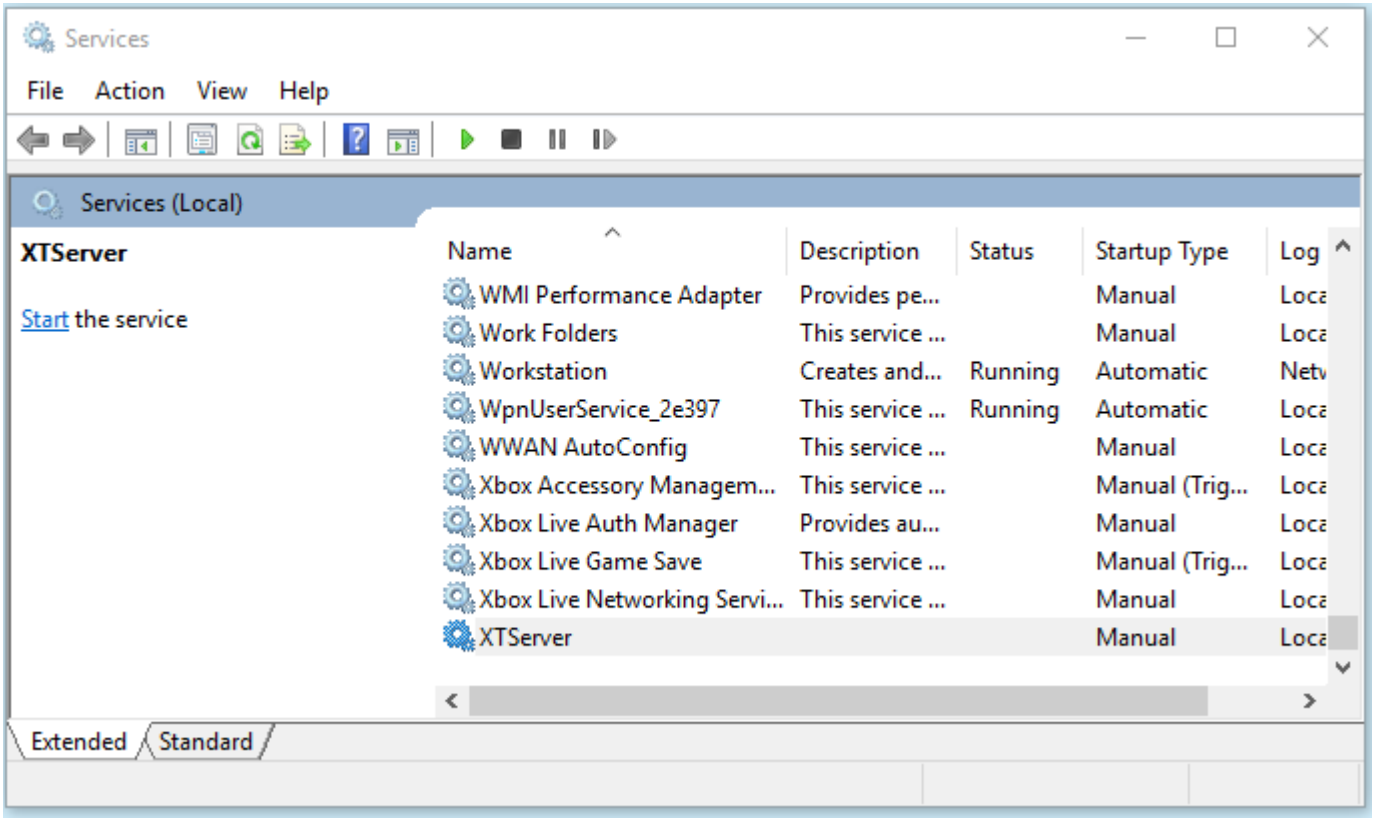
Tool will exit if all jobs stop. Press Ctrl-C on the keyboard if you want to exit “xtrun” manually.

You can also run individual jobs from the list of all jobs. Simply provide job name as first parameter to “xtrun” utility. If job name contains spaces then it has to be enclosed in double quotes.
Jobs that have to finish need “exit” task. Jobs that do not have “exit” task will keep on processing in the loop.

Windows Service

Program also installs Windows Service for automated start and execution. Service XTServer will not auto-start by default. If you want it to auto-start change “Startup Type” to “Automatic” using Windows Services screen (part of Windows OS).
Before you try XTServer service please try running XT Server via command line. It is easier to find configuration issues using interactive command prompt.

XTServer redirects on-screen output into log file **servicelog.txt** when it runs as Windows Service.



Access XTServer service via Windows Services.

Home Directory

If environment variable XTSERVERDIR is set then XT Server will use it to locate and store processing files and subfolders under %XTSERVERDIR%\Documents\xtserver.

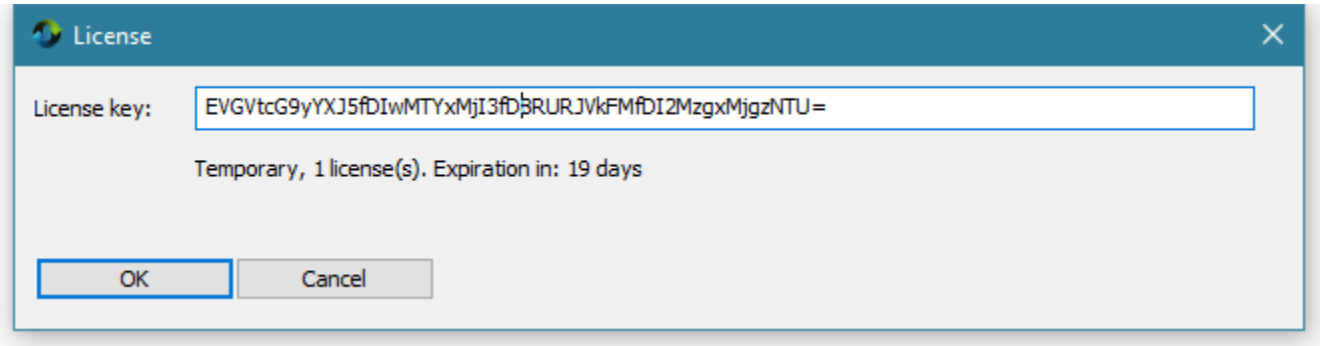
In case if environment variable XTSERVERDIR is not set then %USERPROFILE%\Documents\xtserver will be used.

Change of environment variable may require restart computer.

License

XT Server comes with trial and retail license. You can try and evaluate it free for 3 weeks. Once trail version expires please purchase license for continues use.

To start a trial version and receive license key for 3 weeks simply run GUI tool and open Help-License screen.



License key entered in the GUI will be also used by command line tools.