TermImport: SFTP support

Description

TermImport plugin is able to watch a remote filesystem folder for terminology-to-be-imported to translate5.

Remote SFTP folder will be queried every ~15 min with the usual translate5 cron job

The plug-in is able to watch multiple folders e.g. located on different SFTPs at the same time. This is implemented by ability to overwrite global SFTP settings on customer-level.

All terminology is imported with respect to a config, which should be copied from **TermImport/config/instruction.ini** and uploaded to the remote SFTP directory's Import-folder, which is then accessed by translate5.

If there is no **instruction.ini** in the remote directory's Import-folder - the import won't be started.

Folder structure

Each remote SFTP directory may have from 1 to 2 of 4 possible folders (sub-directories):

- **Import** - This is single folder, initially expected to exist inside SFTP directory. It, in it's turn, is expected to have:
  - ***.tbx** files to be imported
  - **instruction.ini** that contains the settings to control the import behaviour of *.tbx-files
- **Error**
  - If at least one tbx-file was failed to import - this folder will be automatically created and failed tbx-file will be moved here
- **Import-running**
  - If we initially had **Import** folder having valid **instruction.ini** file then this folder is renamed from **Import** to **Import-running**, so this is indicating that import process is started
- **Import-success**
  - Once at least one tbx-file has been successfully imported - this folder is auto-created and this tbx-file is moved here. Still the import itself may running in translate5.

Example structure of an remote directory, that containts a terminology files to be imported:

/some/dir/with/tbx

  /Import

    /instruction.ini

    /terminology1.tbx

    /terminology2.tbx

How it looks after being moved to Import-success:

/some/dir/with/tbx

  /Import-success

    /instruction.ini

    /terminology1.tbx

    /terminology2.tbx

Instruction.ini

The information in the instruction.ini is evaluated as explained in the attached instruction.ini file.

For all parameters that are not present in the copy of instruction.ini that you'll upload to your remote sftp directory - the ones shown below are assumed as default

```ini
; If this is true, terms will be merged as described at https://confluence.translate5.net/display/TAD
/Term+Collection.
; Otherwise all terms will be added to the termCollection, if the ID in the TBX does NOT exist already in the DB
; when the ID of the termEntry of the TBX does already exist in the DB, the terms of this termEntry will always
be merged
; Possible values: true or false.
mergeTerms=true

; Deletes all terms in all listed termCollections, that have a modification date older than the listed one.
; Since every term that exists in a TBX gets a new updated date on TBX-import, even if it is not changed:
Simply set this date to yesterday to delete all terms, that are not part of the current import
; The updated date is a date internal to translate5 and different from the modified date of the term, that is
shown in the interface
deleteTermsLastTouchedOlderThan=2018-05-01

; Deletes all terms in all listed termCollections, that have a modification date older than the start of the
current import.
; Possible values: true or false
deleteTermsOlderThanCurrentImport=true

; Delete all proposals in a listed termCollections, that have a creation date older than this configs value.
deleteProposalsLastTouchedOlderThan=2018-05-01

; Delete all proposals in a listed termCollections, that have a creation date older than the start of the
current import.
; Possible values: true or false
deleteProposalsOlderThanCurrentImport=false

; Mappings between tbx-file names, expected to exist in remote sftp directory and local TermCollection names
; where those files shoud be imported into. If some TermCollection does not exist so far - it will be created
; if there the customer exists having the specified number (see comments for further [CollectionMapping]
section)

[FileMapping]
Term1.tbx=exampletermCollection
Term2.tbx=exampletermCollection2
;Term3.tbx=exampletermCollection2

; Mappings between TermCollection names and customer numbers.
; If there is no customer having such number in the database
; then tbx file won't be imported into such a TermCollection
[CollectionMapping]
exampletermCollection3=10001
exampletermCollection2=123456789
exampletermCollection=123456789
```

API point for manual start

Users with role **api** have right to manually start processing.

Endpoint: **editor/termimport/force-check**

Possible GET params:

Param name	Values	Description
none	-	If no params provided - all clients and default configs will be processed
defaultOnly	true|false	Only default config will be processed
clientIds	integer[]	Comma separated client ids. Only provided client ids will be processed (if some present of course)

Configuration

Config name	Values	Default	Description
runtimeOptions.plugins.TermImport.filesystemConfig	map	[]	Filesystem config for terminology import from remote SFTP directory

FilesystemConfig explanation

Default config in **PreferencesSystem configuration** always checked

If it is set for any customer then except from **Default** configuration customer's one will also be checked.

So in case with **filesystemConfig** setting value in customer entry not over-write system's but extends it instead.

Config itself is a simple JSON object.

Example:

```
{"host": "sftp-server","username": "translate5","password": "translate5", "type": "sftp", "rootpath": "/some/dir
/with/tbx"}
```

Field name	Values	Variants	Local Type	Required	Default	Description
type	string	• sftp		+		Type of filesystem
linkHandling	string	• 0001 • 0002	+		0002	How to deal with links, either **0001** (skip) or **0002** (disallow) Disallowing them causes exceptions when encountered
writeFlags	integer	• 1 • 2 • 3	+		2	• **1** to acquire a shared lock (reader). • **2** to acquire an exclusive lock (writer). • **3** to release a lock (shared or exclusive)
host	string			+		sftp host
username	string			+		sftp login
password	string					sftp password set to null if privateKey is used
privateKey	string					can be used instead of password, set to null if password is set must be a path: '/path/to/my/private_key' if a relative path is given, APPLICATION_ROOT is prepended
passphrase	string					set to null if privateKey is not used or has no passphrase
rootpath	string				/	the root directory to be used on the SFTP server
port	integer				22	SFTP server port
useAgent	bool				false	
timeout	integer				10	
maxTries	integer				4	
hostFingerprint	string					