Allgemein

Hilfe-Output

Um die Parameter einer Operation zu sehen, kann die entsprechende Klasse ohne Parameter auf der Shell aus dem Verzeichnis %FEEDER_JAVA% aufgerufen werden, bspw. SIP-Extractor:

```
java ch.docuteam.feeder.qualityassurance.SIPExtractor
```

Ergibt folgenden Output:

```
ERROR 2014-01-02T10:54:25.140 (SIPExtractor) A wrong number of parameters
was passed to the command executor
INFO 2014-01-02T10:54:25.140 (SIPExtractor) Usage: java
ch.docuteam.documill.qualityassurance.SIPExtractor [path/to/]SIP
Parameters:
        [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the docudarc.workbench.workdir ...
```

Admin

Version

The Version class returns the current version number of the docuteam feeder library and the version numbers of the docuteam libraries it depends on.

```
Usage: java ch.docuteam.feeder.admin.Version -v
docuteam feeder: 2.4.9 (17.04.2015)
docuteam darc: 2.14.2 (10.03.2015)
docuteam converter: 1.0.4 (10.03.2015)
docuteam tools: 1.10.6 (25.02.2015)
AIPCreatorETH: 1.0.0i (17.02.2015)
```

Ingest

BARSIPConverter

The BARSIPConverter converts a BARSIP SIP to a SIP, that the Matterhorn Profile.

```
Usage: java ch.docuteam.feeder.ingest.BARSIPConverter [path/to/]BAR-SIP
[targetFolder]
Parameters:
    [path/to/]BAR-SIP: name of the BAR-SIP folder or ZIP-file; if no path is
```

```
given, it will be expected to be in the location defined by the
'feeder.workbench.dropbox' property
    [targetFolder]: directory where to move the created SIP to; if omitted,
the SIP will be moved to the location defined by the
'feeder.workbench.workdir' property
```

CheckWorkbenchSpace

The CheckWorkbenchSpace checks whether the workbench has the necessary space available for the processing of the given SIP. By default, it calculates with three potential copies for a given SIP on the workbench at the same time.

```
Usage: java ch.docuteam.feeder.ingest.BARSIPConverter [path/to/]BAR-SIP
[targetFolder]
Parameters:
    [path/to/]BAR-SIP: name of the BAR-SIP folder or ZIP-file; if no path is
given, it will be expected to be in the location defined by the
'feeder.workbench.dropbox' property
    [targetFolder]: directory where to move the created SIP to; if omitted,
the SIP will be moved to the location defined by the
'feeder.workbench.workdir' property
```

Cleanup

The Cleanup is used to delete a given SIP from the work folder in the workbench. It will use the workbench defined in the `docuteamFeeder.properties` file. Optionally, also the preparation directory of the workbench will be cleaned from any SIP version of the same name.

```
Usage: java ch.docuteam.feeder.ingest.Cleanup [path/to/]SIP [prep]
Parameters:
    [path/to/]SIP: name of the SIP. If not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [prep]: if 'true', SIPs of the same name in the preparation folder will
be removed as well; defaults to 'false'
```

CreateEADFile

The CreateEADFile operation creates a single EAD file from the EAD chunks of each node of a given SIP and puts it into the `finished` directory

```
Usage: java ch.docuteam.feeder.ingest.CreateEADFile [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

ExtentCalculator

The ExtentCalculator will calculate the amount of files within each folder node and assign its value to the respectives folder element 'extent'. 'material' is set to 'Datei(en)'.

```
Usage: java ch.docuteam.feeder.ingest.ExtentCalculator [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

SIPFileMigrator

The SIPFileMigrator compares the files of a SIP with the settings of a configuration file (migration-config.xml) and converts the files according to the definitions in that file.

```
Usage: java ch.docuteam.feeder.ingest.SIPFileMigrator [path/to/]SIP
keepOriginals
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    keepOriginals: { true | false }, indicating whether to keep the original
files after the migration process
```

SIPRemoveFromDropbox

The SIPRemoveFromDropbox operation will remove a given SIP from the dropbox to an indicated folder or deletes it if no target folder is given.

```
Usage: java ch.docuteam.feeder.ingest.SIPRemoveFromDropbox [path/to/]SIP
[targetFolder]
Parameters:
    [path/to/]SIP: path of the SIP; if only the name is given, it will be
expected to be in the location defined by the feeder.workbench.dropbox
property
    [targetFolder]: directory where to move the SIP to; if omitted, the SIP
will be deleted
```

Quality Assurance

FilePathLengthCheck

Check if any canonical filepath within a given folder exceeds a given number of allowed characters.

Any file or folder exceeding the maximal allowed path lengh will be logged.

```
Usage: java ch.docuteam.feeder.qualityassurance.FilePathLengthCheck
/absolute/path/to/folder maxAllowedFilePathLength
Parameters:
    /absolute/path/to/folder: absolute path of the folder that should be
checked
    maxAllowedFilePathLength: the max allowed number of characters of the
canonical file path
```

SIPConfirmation

Connect to the Fedora repository and get a single PID to identify the SIP.
In the sequence, this PID will be used as the main entry point in the repository for the submission. The value will be stored in the <mets:OBJID> element.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPConfirmation
[path/to/]SIP [PIDNamespace[:###]]
Parameters:
    [path/to/]SIP: name of the SIP. If not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [PID namespace[:###]]: namespace for new PID or complete PID to use for
the object; if omitted, the standard namespace from the submission agreement
will be used; if the submission agreement cannot be found, the default
namespace of the Fedora repository will be used.
```

SIPConvertToSafeFileNames

Rename files containing special characters.
Safe filenames contain only the characters A-Z, a-z, 0-9, and „_.-“.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPConvertToSafeFileNames
[path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

SIPDeleteBackupFiles

Delete backup files of a SIP.
A list of filename patterns must be supplied to specify which files to delete.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPDeleteBackupFiles
[path/to/]SIP  [filenamePattern filenamePattern ...]
Parameters:
```

```
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [filenamePattern filenamePattern ...]: a list of filename patterns (NOT
case-sensitive, '*' is wildcard, but is only allowed at the beginning or end
of the pattern). Files matching any one of this patterns will be deleted
```

SIPExtractor

Extract a zipped SIP into the work folder of the workbench.
The optional second argument can be used to indicte a different target folder.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPExtractor [path/to/]SIP
[targetdir]
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [targetdir]: target directory; absolute path of the directory where to
unzip the SIP to. Optional, default to the workdir defined in the
docuteamFeeder.properties
```

SIPFixityCheck

Check the files contained for conformance with the checksums in the METS file.
The results of the check will be written in the form of PREMIS events as inline xml code into the METS file.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPFixityCheck [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

SIPPathLengthCheck

Check if any canonical file path within the SIP exceeds the maximal allowed path length.
Any file or folder exceeding the maximal allowed path lengh will be logged.

```
Usage: java ch.docuteam.feeder.qualityassurance.SIPPathLengthCheck
[path/to/]SIP  maxAllowedFilePathLength
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    maxAllowedFilePathLength: the max allowed number of characters of the
canonical file path
```

SIPSubmissionAgreementCheck

Check the files formats contained for compliance with the submission agreement.
There are two modes: In the first mode (removeBadFiles = false), any files not complying with the submission agreement will be listed (using WARN log entries) and an error code will be returned.
In the second mode (removeBadFiles = true), any files not complying with the submission agreement will be deleted from the SIP, and the modified METS.xml will be saved (the original SIP remains untouched as a backup).

```
Usage: JAVA ch.docuteam.feeder.qualityassurance.SIPSubmissionAgreementCheck
[path/to/]SIP [removeBadFiles]
Parameters:
    [path/to/]SIP: name of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [removeBadFiles]: optional, { true | false }; indicating whether to
automatically remove files that are not valid according to the submission
agreement
```

SIPVirusCheck

Check each file of the SIP for viruses using a ClamAV (www.clamav.net).
A running clamav daemon is required. Depending on the second argument, it will either throw exceptions or automatically delete infected files.

<ocde>Usage: java ch.docuteam.feeder.qualityassurance.SIPVirusCheck [path/to/]SIP deleteInfected
Parameters:

```
[path/to/]SIP: name of the SIP; if not path is given, it will be expected to
be in the location defined by the feeder.workbench.workdir property
deleteInfected: if 'true', the operation automatically removes infected
files from the SIP</code>
```

Storage

ChecksumChecker

The ChecksumChecker will check the objects with ORIGINAL datastreams in fedora against the generated checksum.
It will use the FEEDER_JAVA system variable to locate configuration files

```
Usage: java ch.docuteam.feeder.storage.ChecksumChecker recipient [namespace]
[namespace] ...
Parameters:
     mailto:recipient@email.com
Optional Parameters:
```

```
    [namespace] [namespace] ...: Fedora namespaces separated by space. If no
namespace is given, all datastreams are checked.
```

DIPDeliverer

The DIPDeliverer gets the datastream(s) for a provided fedora PID or file format PUID, repackages each as a DIP and stores them in a given directory.

```
Usage: java ch.docuteam.feeder.storage.DIPDeliverer['pid'|'puid'] [PID|PUID]
[targetLocation]
Parameters:
     ['pid' | 'puid'] (if 'pid' then provide a fedora PID, if 'puid' then
provide a pronom PUID)
     [PID|PUID] (PID = fedora persistent unique identifier, PUID = pronom
persistent unique identifier)
Optional Parameters:
    [targetLocation]: Location where to save DIP.
```

FedoraObjectUpdater

This operation uploads new versions of object to the fedora server.

```
Usage: java ch.docuteam.feeder.storage.FedoraObjectUpdater [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP. If not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

FOXMLCreator

The FOXMLCreator converts a given METS package into separate FOXML (Fedora Object) files.
It will use the current working directory or - if not available - the FEEDER_JAVA system variable to locate configuration files and the workbench defined in the docuteamFeeder.properties file.
The code makes use of the Directory Ingest tool which is available from the Fedora website.
Modifications were done to support distinctions between root folders and folders through conversion rules (crules.xml) and to be able to handle already gotten PIDs during the ingest process.

```
Usage: java ch.docuteam.feeder.storage.FOXMLCreator [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP. If not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
```

FOXMLIngester

The FOXMLIngester will transfer a given list of FOXML (Fedora Object) files to a Fedora repository for storage.
It will use the FEEDER_JAVA system variable to locate configuration files and the workbench defined in the docuteamFeeder.properties file.

```
Usage: java ch.docuteam.feeder.storage.FOXMLIngester [path/to/]SIP
[keepFOXML]
Parameters:
    [path/to/]SIP: name of the SIP. If no path is given, it will be expected
to be in the location defined by the feeder.workbench.finished property
    keepFOXML: One of { true | false }, indicating whether to keep the FOXML
files after a successful ingest; defaults to 'true'
```

METSValidator

The METSValidator validates the mets xml file with the linked schema definitions and places the namespace declarations from the root element to the respective elements.
This is a necessary preparation for the mets xml when it has to be split up into different parts, as is the case when several foxml files are being created out of the SIP (FOXMLCreator).

```
Usage: java ch.docuteam.feeder.storage.METSValidator [path/to/]SIP [withEAD]
Parameters:
    [path/to/]SIP: path of the SIP; if not path is given, it will be
expected to be in the location defined by the feeder.workbench.workdir
property
    [withEAD]: whether to include EAD as descriptive metadata and create a
datastream in the fedora objects; defaults to false
```

PIDAssigner

The PIDAssigner will obtain PIDs from fedora and assign them to the nodes in the sip.
It will use the FEEDER_JAVA system variable to locate configuration files and the workbench defined in the docuteamFeeder.properties file.

```
Usage: java ch.docuteam.feeder.storage.PIDAssigner [path/to/]SIP
Parameters:
    [path/to/]SIP: name of the SIP. If no path is given, it will be expected
to be in the location defined by the feeder.workbench.workdir property
```

PIDListPublisher

The PIDListPublisher saves/sends the file 'PIDs.txt' which results from the FOXMLIngester class to a given URL.

```
Usage: java ch.docuteam.feeder.storage.PIDListPublisher [path/to/]SIP
receiverURL
Parameters:
```

```
    [path/to/]SIP: path of the SIP; if no path is given, it will be expected
to be in the location defined by the feeder.workbench.finished property
    receiverURL: An URL in the style of { file: | mailto: }, indicating
whether to put/send the list with PIDs.
```

RenameSIPasAIPforIaas

The operation RenameSIPasAIPforIaas renames an SIP using the pid of the root element of the mets file as prefix.

```
Usage: java ch.docuteam.feeder.storage.RenameSIPasAIPforIaaS [path/to/]SIP
[targetFolder]
Parameters:
    [path/to/]SIP: path of the SIP; if only the name is given, it will be
expected to be in the location defined by the feeder.workbench.dropbox
property
    [targetFolder]: directory where to put the AIP to; if omitted, the AIP
will be copied to standard output directory '4_output'
```

UpdateExcelWithPID

This class will write PIDs from an SIP's nodes into excel sheet(s). The excel sheet(s) must have a column with a label of either 'identifier' or 'id' in the first row. PIDs will be written into the column with the header string 'PID' or – if such a column is not available – into the next free column.

```
Usage: java ch.docuteam.feeder.storage.UpdateExcelWithPID [path/to/]SIP
path/to/folder/with/excel
Parameters:
    [path/to/]SIP: path to the SIP; if relative path is given, try to find
it in the workbench's working directory
    path/to/folder/with/excel: path to the excel files to be updated
```

Submission

AgreementsOverviewGenerator

Creates a simple overview of submission agreements located in a given folder. This is done by XSL transformations, for which the class looks for any submission agreement files in the given directory and lists them in a simple xml structure:

```
<safilelist> \\
  <sa_1 /> \\
  <sa_2 /> \\
  ... \\
  <sa_x /> \\
```

```
</safilelist>
```

```
Usage: java ch.docuteam.feeder.submission.AgreementsOverviewGenerator
agreements_directory type output_directory
Parameters:
    agreements_directory: location where the collection of submission
agreements can be found
    type: one of { Hierarchy | Flat | CSV }, defining the structure of the
resulting overview file
    output_directory: target location for the created overview file,
defaults to the directory, where the agreements are located (args[0])
```

CheckFolder

Check if sip size and size of each file in folder and file paths within the SIP exceeds the maximal allowed provided value.
Any file or folder exceeding the maximal allowed size will be logged.

```
Usage: java ch.docuteam.feeder.submission.CheckFolder [/path/to/]folder
maxTotalSize maxSingleFileSize maxFilePathLength
Parameters:
    [/path/to/]folder: path of the folder to check; if not path is given, it
will be expected to be in the location defined by the
feeder.workbench.workdir property
    maxTotalSize: the max allowed size the folder may have
    maxSingleFileSize: the max allowed size any of the files contained by
the sip
    maxFilePathLength: the max allowed length of file paths within the
folder
```

CreateSIPsFromFileOrFolder

The CreateSIPsFromFileOrFolder operation will create SIPs from a given file or folder. If the source is a folder, a parameter will define whether a single SIP or separate SIPs for each child should be created.

```
Usage: java ch.docuteam.feeder.submission.CreateSIPsFromFileOrFolder source
recursive saID dssID author [dropbox]
Parameters:
    source: file or folder for which an SIP should be generated
    split: if 'true', a separate SIP will be created for each file/folder
within the source (assuming the source is a folder)
    saID: value to use for referencing a submission agreement in the SIP
    dssID: value to use for referencing a data submission session of the
respective submission agreement
    author: value to use as the creator for the SIP
    [dropbox]: optional location where to put the SIPs; if omitted the
property 'feeder.workbench.dropbox' defined in the docuteamFeeder.properties
will be used
```

SubmitSIPsFromFolder

The SubmitSIPsFromFolder will use the given arguments for selecting SIPs in a folder and submitting them to a number of workflows using the feeder REST-interface.

```
Usage: java ch.docuteam.feeder.submission.SubmitSIPsFromFolder dropbox
filter feeder_url workflows user password useAbsolutePaths checkEmptyQueue
[maxNumberSIPs]
Parameters:
    dropbox: path to the folder containing the SIPs
    errorbox: path to the folder where to put unsuccessful SIPs
    filter: regex filter string for the SIPs within the dropbox; put the
regex expression into quotation marks!
    feeder_url: URL pointing to the feeder main page, f.ex.
http://localhost/feeder
    workflows: comma separated list of workflows to execute on each SIP
    user: username for feeder
    password: password for feeder
    useAbsolutePaths: true/false, indicating whether to submit SIPs by
absolute paths or just their filenames
    checkEmptyQueue: true/false, indicating whether to check if the queue is
empty before submitting new SIPs
    [maxNumberSIPs] (optional): maximum number of SIPs to send to feeder; if
omitted, all SIPs matching the filter string will be submitted
```

WebjaxeAgreementCollector

The WebjaxeAgreementCollector will look for the any submission agreements created within the webjaxe editor and copy them to a given directory.

```
Usage: java ch.docuteam.feeder.submission.WebjaxeAgreementCollector
target_directory [webjaxe_home]
Parameters:
    target_directory: location where to store the submission agreements xml
files
    [webjaxe_home]: optional location of the webjaxe installation directory,
usually in the webservers htdocs directory. If omitted, the environment
variable $WEBJAXE_HOME will be used.
```

Util

MailSender

This MailSender operation sends an email to the given recipient with optional attachments.

```
Usage: java ch.docuteam.feeder.util.MailSender receiver subject text
```

```
[attachment1 [attachment2 [...] ] ]
Parameters:
    receiver: the receiver's e-mail address(es), comma separated if several
    subject: the mail subject
    text: the message text
    attachments: filepaths to attachments; if the first attachment is 'zip',
all attachments will be zipped into a single attachment
```