Folder apps

Description

This is a bundle of six apps that do interesting things with folders.

- **Filetypes Counter** counts the files in a job folder per filetype, and adds these counters and some extra info to the job folder as private data.
- **Delete Files** deletes files from a job folder.
- **Hot Folder Monitor** checks a certain folder every so often to see if files disappear from the folder. If not, it means that the application that should be processing those files is not running.
- **File Pool Cleanup** allows to remove files from a file pool that are older than a certain number of days.
- **File Property Sort** puts files inside a job folder in a certain alphabetical order based on a property of the file.
- **Submit Filespec** can be used to submit files from a hierarchy, but only those that match a certain file specification.

Compatibility

Switch 2018 and higher. Windows or Mac OSX.

Compatibility third-party applications

There are no dependencies on third-party applications.

Filetypes Counter

Filetypes Counter counts the files in a job folder per filetype, and adds these counters and some extra info to the job folder as private data.

For each filetype 3 private data will be added: the count, the names and the paths of the files (separated by *).

The name of the private data is built like this: <key>.<filetype>.count
<key>.<filetype>.names
<key>.<filetype>.paths

Private data will always be added for all the filetypes you define in the properties, even if there is no file of that filetype, so that you can use this information further in your flow.

The other files will be counted together in <key>.others.count, <key>.others.names and <key>.others.paths, unless you choose to add separate private data for each filetype found. An extra 'noExtension' filetype will be added if necessary.

Depending on your OS, some invisible files can be present in the job folder, so these special filetypes are always counted as well: startingWithDot, .DS_Store, and thumbsdb.

Extra stats private data will always be added:

- <key>.folders.count: number of subfolders found
- <key>.folders.paths: paths of subfolders found, separated by *
- <key>.stats.count: count of all files
- <key>.stats.hidden: count of startingWithDot, .DS_Store, and thumbsdb files
- <key>.stats.visible: count - hidden
- <key>.stats.folderLevels: number of sublevels found in the job folder

Connections

Filetypes Counter has incoming connections and a single outgoing connection.
Jobs that are not folders pass unchanged.

Properties detailed info

Properties

Property	Value
Name	Filetypes Counter
Description	
Subfolder levels	All
Filetypes	Multi-line text with variables defined
Count other found filetypes	Add separate private data
Private data key prefix	filetypes

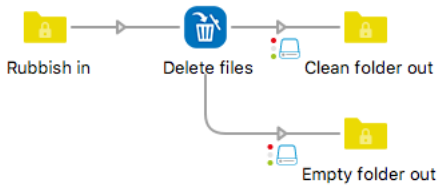
Flow element properties

- **Subfolder levels:** Which subfolder level(s) should be scanned?
  All, Root level only or Custom
  If Custom: From (top level), *considering root level = 1*
  To (bottom level), *must be greater than or equal to From (top) level*
- **Filetypes:** Select the filetypes to count.
  If custom, one per line, format must be of type *.pdf
- **Count other found filetypes:**
  - Together (others)
  - Add separate private data
- **Private data key prefix:** name of private data key prefix, default 'filetypes'

Sample result

filetypes.startingWithDot.count	5
filetypes.startingWithDot.names	.AdobeFnt13.lst*.AdobeFnt13.lst*.AdobeFnt13.lst copy*.AdobeFnt13.lst copy*.hidden pdf.pdf
filetypes.startingWithDot.paths	/dotfiles/.AdobeFnt13.lst*/dotfiles/dotfiles bis/.AdobeFnt13.lst*/dotfiles/dotfiles bis/.AdobeFnt13.lst copy*/dotfiles/.AdobeFnt13.lst copy*/B/BB/.hidden pdf.pdf
filetypes.DS_Store.count	4
filetypes.DS_Store.names	.DS_Store*.DS_Store*.DS_Store*.DS_Store
filetypes.DS_Store.paths	/.DS_Store*/dotfiles/.DS_Store*/A/.DS_Store*/B/.DS_Store
filetypes.thumbsdb.count	1
filetypes.thumbsdb.names	Thumbs.db
filetypes.thumbsdb.paths	/B/Thumbs.db
filetypes.pdf.count	20
filetypes.pdf.names	00008.pdf*00012.pdf*00013.pdf*00011.pdf*00014.pdf*00015.pdf*00006.pdf*00007.pdf* 00005.pdf*00004.pdf*00001.pdf*00003.pdf*00002.pdf*00009.pdf*00010.pdf*00024.pdf* 00025.pdf*00021.pdf* 00022.pdf*00023.pdf
filetypes.pdf.paths	/00008.pdf*/A/00012.pdf*/A/00013.pdf*/A/00011.pdf*/A/00014.pdf*/A/00015.pdf*/00006.pdf* /00007.pdf*/00005.pdf*/00004.pdf*/00001.pdf*/00003.pdf*/00002.pdf*/B/BB/00009.pdf* /B/BB/00010.pdf* /B/00024.pdf*/B/00025.pdf*/B/00021.pdf*/B/00022.pdf*/B/00023.pdf
filetypes.indd.count	0
filetypes.indd.names	-
filetypes.indd.paths	-
filetypes.idml.count	2
filetypes.idml.names	calendar 2020-2021.idml*calendar 2019-2020.idml
filetypes.idml.paths	/B/calendar 2020-2021.idml*/B/calendar 2019-2020.idml
filetypes.png.count	2
filetypes.png.names	Filetype-Counter-icon-32.png*Filetype-Counter-icon-32.png
filetypes.png.paths	/Filetype-Counter-icon-32.png*/B/BB/Filetype-Counter-icon-32.png
filetypes.xml.count	6
filetypes.xml.names	cdCollection.xml*customer.xml*inventory.xml*food.xml*l_01.xml*P2P51x_reportOPT.xml
filetypes.xml.paths	/A/cdCollection.xml*/A/customer.xml*/B/inventory.xml*/B/food.xml*/B/l_01.xml* /B/P2P51x_reportOPT.xml
filetypes.noExtension.count	2
filetypes.noExtension.names	no extension 1*no extension 2
filetypes.noExtension.paths	/no extension 1*/B/BB/no extension 2
filetypes.others.count	0
filetypes.others.names	-
filetypes.others.paths	-
filetypes.folders.count	5
filetypes.folders.paths	/dotfiles/*/dotfiles/dotfiles bis/*/A/*/B/*/B/BB/
filetypes.stats.count	42
filetypes.stats.hidden	10
filetypes.stats.visible	32
filetypes.stats.folderLevels	3

Delete Files



Delete Files app is one that deletes certain files from a job folder. This can be done by specifying the files that have to be kept, for example, keep all PDF files and delete the rest, or inversely by specifying the files that have to be deleted, for example, delete all files that do not start with a number.

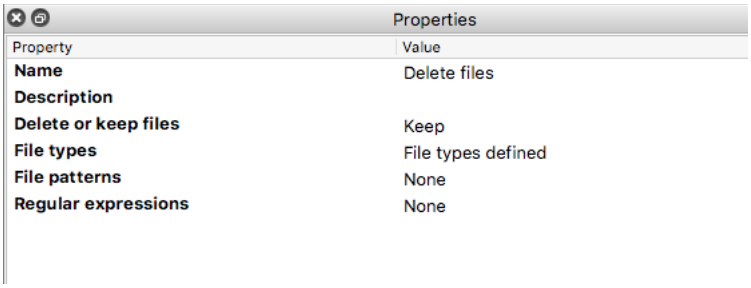
The specification of files that have to be kept or deleted can be done in three ways: by file types, by file patterns, or by regular expressions. The three methods can be combined.

When the job is not a folder it is ignored and passed along the outgoing connection.

Connections

Delete Files has outgoing traffic-light connections: Success and Error. The error connection is used in case the criteria you have defined for deleting the files would delete ALL files from the job folder. This of course does not make sense. In all other cases, the job folder is sent along the success connection, also when no files have been deleted.

Properties detailed info



Flow element properties

- **Delete or keep files**: this is a drop-down of “Delete” and “Keep”. It determines whether the files matched by the properties below will be deleted or kept. The default is “Delete”.

- **File types**: here you define the file types that have to be deleted or kept using the file types dialog. Default: None.

- **File patterns**: here you can define file patterns using wildcards, ? standing for 1 unknown character, and * for any number of unknown characters. There can be multiple file patterns, one per line. Default: None.

- **Regular expressions**: this property works in the same way as the file patterns property, except of course that here you define multiple regular expressions, one per line. Default: None.

The three methods can be combined. When the “Delete” option is chosen a file is marked for deletion as soon as it matches one of the criteria. When the “Keep” option is chosen a file is marked for deletion if it matches none of the criteria.

Hot Folder Monitor

Hot Folder Monitor checks a certain folder every so often and remembers the list of files that are present in that folder. When there are files that have not disappeared from that folder the next time around a text file is created listing those files. This text file can then be used to trigger a mail to alert somebody to the fact that the application that should be processing the files from the folder may not be running. The app also sets the e-mail body of the text file job to the same contents. This is convenient if you prefer to put the list of files in the mail body rather than attach the text file.

When the app sees that all files have disappeared, no text file is created.

The app only checks files. It ignores subfolders of the folder that is being monitored.

Connections

Hot Folder Monitor does not have any input connections. It has a single output connection along which the text file is sent when necessary.

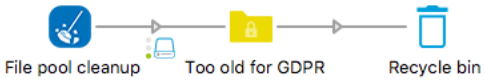
Properties detailed info

Flow element properties

- **Hot folder to be monitored**: choose the folder to be monitored. There is no default.

- **Interval (secs)**: make sure to choose a value that is higher than that used by the application that scans the monitored folder. Also allow for some extra processing time to reduce the chance of false alarms. Default: 300. Note that when the application is not running for a longer period of time the app will trigger the notification every time it runs!

File Pool Cleanup



File Pool Cleanup checks a certain root folder every so often and scans it and all its subfolders looking for files that are older than a certain number of days. These files are removed from their location and injected into the flow for further processing, usually deleting them or archiving them in a different location. One use case for this tool is to clean up the location where you store customer files in order to be GDPR-compliant.

The root folder must be a local folder, network folders are not allowed! Also selecting the root folder of a local drive is not allowed. These are measures to avoid that files are being removed from important locations.

Another safety measure is that you can specify the maximum number of files to be processed. When the limit is passed the files are not removed but instead a log file is created containing a list of the files that should have been removed. During the setup phase it is therefore a good idea to set the limit to 1. This will most probably generate a log file that you can analyze to verify you are using the correct settings. When everything is OK, you can set the limit to 0 which stands for no limit.

There is a choice of selecting files based on the modification date or the creation date.

Finally, there is an option to remove empty folders. Note that only folders that are empty when the app runs are removed. Folders that become empty as a result of the cleanup are not immediately removed, but they will be the next time.

Connections

File Pool Cleanup does not have any incoming connections. It has outgoing traffic-light connections, a Data Success one for the files that have been cleaned up from the file pool, and a Log Error one to which the log file is sent in case the limit has been exceeded.

Properties detailed info

Properties

Property	Value
Name	File pool cleanup
Description	
Path	/Switch workspace/Archive file pool
Interval (hours)	24
Remove files older than (days)	30
File date type	Modification date
Remove empty folders	No
Maximum number of files	0

Flow element properties

- **File pool root folder**: choose the root folder to be cleaned up. There is no default.
- **Interval (hours)**: Default: 24.
- **Remove files older than (days)**: Default: 30.
- **File date type**: this is a drop-down of “Modification date” and “Creation date”. Default: Modification date.
- **Remove empty folders**: Yes or No, default: Yes.
- **Maximum number of files**: When the number of files eligible for cleanup is higher than this number, the files are not removed, but logged in a file that is sent along the log error connection. Default: 0.

File Property Sort

There is an element *Sort files in job* that sorts files in a job based on the file name. The sorting is achieved by adding a prefix to the file names inside the job folder. If you have a list of files that are alphabetically ordered like this:
abc_2.pdf
def_4.pdf
ghi_1.pdf
jkl_3.pdf
and you use the number at the end for sorting the files, the result will look like this:
1_ghi_1.pdf
2_abc_2.pdf
3_jkl_3.pdf
4_def_4.pdf
By adding the prefix, the files are also alphabetically sorted in the desired order.

This app works similarly but instead of using the file name, it uses a file property. The properties currently supported are the file creation date, the file modification date, the file size, the number of pages and the page surface of the first page.

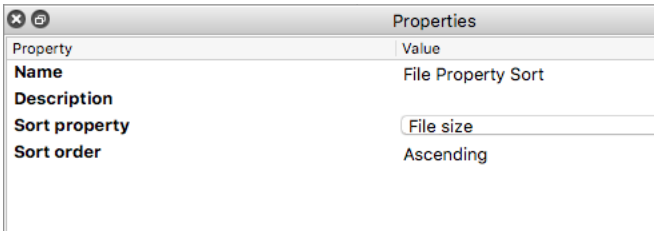
Another difference is that the app allows to sort in ascending or descending order.

The app only scans the root folder of the job. Subfolders, if any, are treated as files. The sorting on the number of pages and the page surface only works for PDF files. Other file types will be treated as having 0 pages and 0 surface, meaning they will come out on top.

Connections

File Property Sort has incoming connections and a single outgoing connection. Jobs that are not folders pass unchanged.

Properties detailed info



Flow element properties

- **Sort property:** a drop-down list of
  - File creation date
  - File modification date
  - File size
  - File name
  - Number of pages
  - Page surface

  Default: File modification date

- Sort order: a drop-down of “Ascending”, “Descending”. Default: Ascending.

Submit Filespec

Submit Filespec works similarly to *Submit hierarchy*, but of course somewhat differently. *Submit hierarchy* allows you to ignore certain folders, but it always processes all the files in the folders that should be processed. This app allows to define file name characteristics for submitting files, eg only PDF files, or only files that start with a number, etc.

The file name specifications can be defined as file types, file patterns, and regular expressions. All three can be used at the same time: when a file matches one of the criteria, it is submitted into the flow.

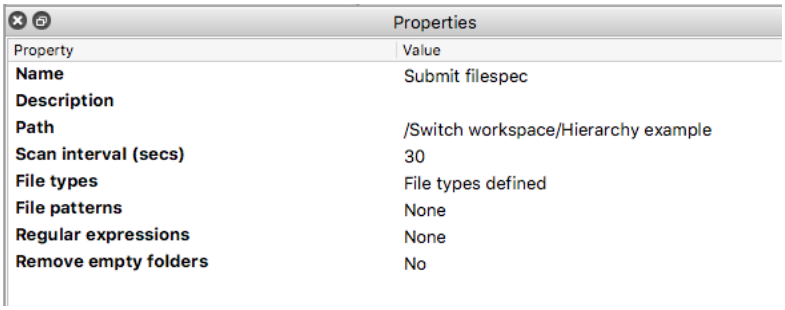
Similar to *Submit hierarchy* this app also sets the hierarchy of the input file.

Finally, there is an option to remove empty folders. Note that only folders that are empty at the moment the app scans the hierarchy are removed. Folders that become empty as a result of all files in the folder having been submitted are not immediately removed, but they will be the next time the app scans the hierarchy and they are still empty.

Connections

Submit Filespec does not have any incoming connections, and it has a single outgoing connection.

Properties detailed info



Flow element properties

- **Path**: choose the root folder of the hierarchy to be scanned. There is no default.
- **Scan interval (secs)**: Default: 30.
- **File types**: here you define the file types that have to be deleted or kept using the file types dialog. Default: None.
- **File patterns**: here you can define file patterns using wildcards, ? standing for 1 unknown character, and * for any number of unknown characters. There can be multiple file patterns, one per line. Default: None.
- **Regular expressions**: this property works in the same way as the file patterns property, except of course that here you define multiple regular expressions, one per line. Default: None.
- **Remove empty folders**: Yes or No, default: Yes.