

# File Management

Plug-In



## Copyright

Copyright © NeuroCheck GmbH  
All rights reserved.  
Version 6.2.5  
Neckarstraße 76-1, 71686 Remseck, Germany

Phone: +49 (0) 7146 - 89 56-0  
Fax: +49 (0) 7146 - 89 56-29  
E-Mail: [info@neurocheck.com](mailto:info@neurocheck.com)  
Web: [www.neurocheck.com](http://www.neurocheck.com)

## Table of Contents

NeuroCheck File Management Plug-In Help - PI_FileManagement.NET.dll .....	3
General Information .....	3
Introduction .....	3
Installation .....	4
Check Functions .....	5
Process File Job .....	5
Introduction .....	5
How to Use .....	6
Parameters .....	8
Parameter Dialog Files .....	8
Parameter Dialog Filter .....	9
Parameter Dialog Processing .....	10
Parameter Dialog Options .....	11
Wait For File Job .....	13
Introduction .....	13
Parameter Dialog .....	14
Support Contact .....	15
About Dialog .....	15
Support Services .....	16

## Introduction

### About NeuroCheck plug-in DLLs in general

A plug-in DLL is a .NET assembly that serves to enhance NeuroCheck with user-defined image processing functionality. The NeuroCheck Plug-In Interface offers the opportunity to integrate user-defined check functions for image processing and data handling. A Plug-In can contain an arbitrary number of self-developed check functions.

These check functions have full access to the NeuroCheck runtime data objects such as Images, ROI Lists and Measurement Lists. The Plug-In check function can be added to a check as well as the built-in standard check functions of NeuroCheck.

Please note that for integration of a plug-in check function into your check routine, a Premium license is required. The completed check routine then can be run with any NeuroCheck license (except the Demo Version).

## Installation

### Installation

Copy the following files from the zip archive to the plug-in directory within the desired NeuroCheck project (e.g. 'C:\Users\Public\Documents\NeuroCheck\6.2\Default\Software Extensions\PlugIns').

- All files inside the `Binaries` directory
- All \*.chm files inside the `Documentation` directory

### Loading a Plug-In

In order to use a Plug-In the Plug-In assembly must be loaded in NeuroCheck. The management of Plug-Ins takes place within the Software Settings dialog. The Software Settings dialog can be found in the System menu of NeuroCheck.

Please note that it is impossible to load or unload a Plug-In as long as a check routine is opened that contains the Plug-In check functions. If the currently opened check routine contains Plug-In check functions then close the check routine first.

Within the Software Settings dialog please select the node Plug-Ins and the sub-node Plug-In DLLs in the tree to the left. The loaded Plug-In assemblies are shown in the List of Plug-Ins. Press the Add button to open a file selection dialog in order to select a further Plug-In assembly.

### Inserting a Plug-In check function to a check routine

A Plug-In check function is inserted using the Check Function Select dialog. All check functions of loaded Plug-Ins are listed in the Plug-In category of the Check Function Select dialog. Within the Plug-In category the check functions are ordered in sub-categories where each sub-category represents the check functions of one Plug-In DLL.

Besides the category the user will hardly notice any difference between the usage of Plug-In check functions and built-in check functions.

## Process File Job: Introduction



### Function overview

This Plug-in check function can be used to move, copy or delete a set of files on disk.

A "File Job" is called a specific configuration of:

1. Files, that should be processed
2. Operation (Movement, Copy or Deletion), that should be applied to the file set.

Each instance of this check function represents one File Job.

To defined the set of files, which should be processed, this check function offers various filter options that can be combined. These are: Filter by name, file type and timestamp. This gives you various possibilities to define your file set.

A File Job can moreover be processed synchronously or asynchronous. An asynchronous processed File Job allows the check routine to continue while the operation on the files is running in the background.

Please check [Process File Job: How to use](#) for further information how to define the file set and the operation.

### Input data

This check function requires no input data object.

### Output data

The check function has no output data object.

### Result view

The check function has no result view.

In case of an error, the error message is displayed in the result view.

### Properties



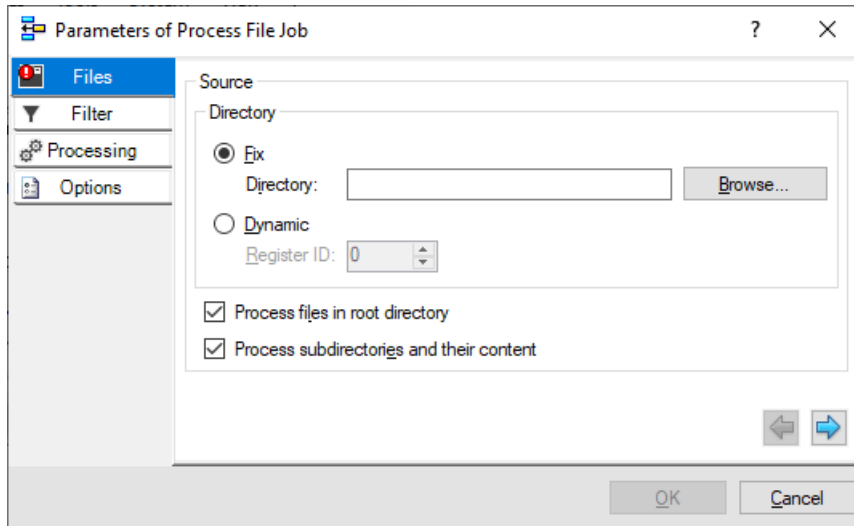
Check function group Plug-In.



The check function has a Parameter Dialog.

## Process File Job: How to Use

To define your File Job, the parameter dialog of this check function is designed like a wizard, that guides you through all options that have to be configured. The following illustration shows the parameter dialog of the check function. There are tab-pages on the left which can either be selected directly or you can step through the single steps with the help of the blue arrows. The parameter dialog also identifies invalid configurations and marks the tab pages that have to be adopted with a red icon, like on the "Source" tab page in the illustration.



### 1. Define the file set

You define the set of files that have to be processed in the tab pages "Files" and "Filter". The idea is to first define a start set of files, by naming a directory and filter the start set afterwards. There are various ways to filter files and folders:

#### 1. Filter by Name:

You can use a filter to only include files by name, for example all images that end in ".png". Or you can remove files from the start set with the exclusion filter by name. Use \* as wildcard. For example NIO\_\*.png. You can provide multiple names that have to be separated by a semicolon.

#### 2. Filter by Date:

This filter can be used if you are interested in the files and folders from a certain timespan, for example all files and folders should be processed that are older than a defined time.

#### 3. Filter by Quantity:

Use this filter if you want to process the oldest files first and ignore a specified amount of newest files.

### 2. Define the target and use buffer directory if needed

You also define the target directory, to which the file set will be processed.

It is possible to define a buffer directory, which will be used if the target directory is not available (or it can not be created) when the check function is executed. This can be the case if it is for example on a network drive or on a USB-device that is not connected. In this case it is advised to use this buffer directory, which should be a location that will definitely be available without external influences. The files and folders will first be copied/moved to the buffer directory. There is a mechanism that checks periodically if the target directory exists (or it can be created) and the operation will be executed if it is available. After a successful processing from the buffer to the target directory, the buffer directory will be cleared.

### 3. Define the Operation

Possible operations are: Copy, Move or Delete. You can define the operation in the tab page "Processing".

### 4. Define Options e.g. asynchronous execution

In the tab page "Options" you can choose if you want to process the file set synchronously or asynchronously. Copying or moving a lot of big files can take a long time and if you wait for the result in your check function, it may take seconds or minutes for which NeuroCheck has to wait. That's why you can start the job asynchronously, so the check function starts the job in the background. The check result will be OK, if the job could be started without problems.

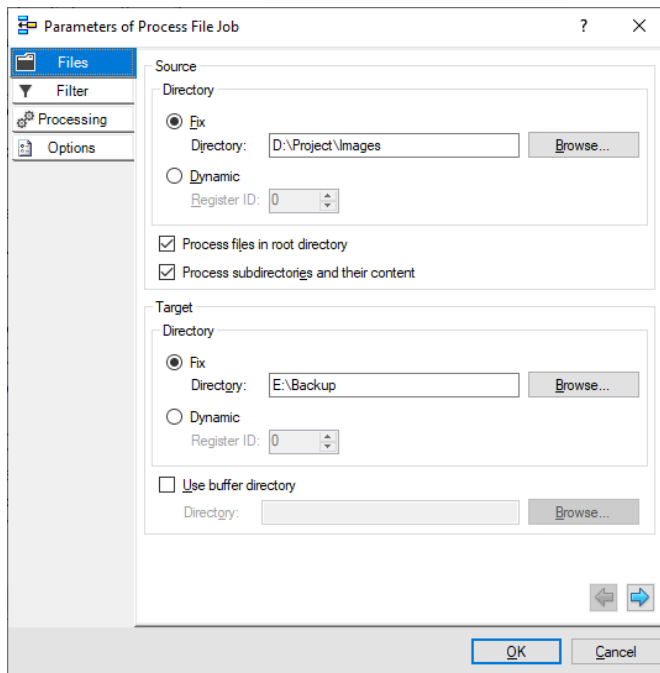
If you want to check if an asynchronous File Job is finished, specify a job identifier in the "Invocation" page and use the "Wait for File Job" check function. It lets you enter a job identifier, use the one you have defined in this check function.

The "Wait for File Job" check function will wait until the job is finished. It is advised to use a timeout, otherwise the check function will

wait forever if the job never finishes.

## Process File Job: Parameter Dialog Files

▾ Screenshot of Parameter Dialog



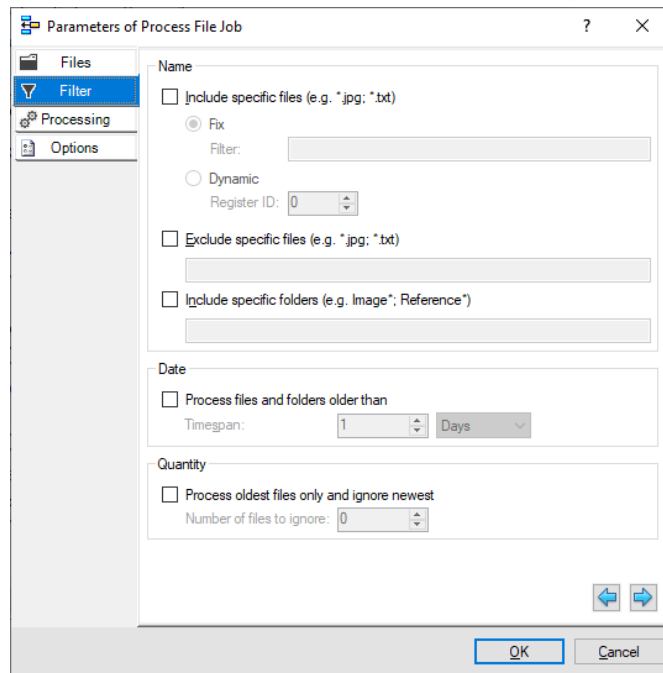
The **Files** tab contains the following elements:

Element	Description
Directory	Directory from which to start the file operations.
Fix	Fix directory path.
Dynamic	Directory path is read from a string register.
Process files in root directory	Check if the files in the root directory should be processed. This does not affect the files in folders.
Process folders and content	Check if folders and their subfolders and files should be processed.
Directory	Directory put the processed files/folders into.
Fix	Fix target directory path.
Dynamic	Directory path is read from a string register.
Use buffer directory	Check if you want to use a buffer directory if the target directory is not available.
Directory	Buffer directory for temporary storage.



## Process File Job: Parameter Dialog Filter

☑ Screenshot of Parameter Dialog

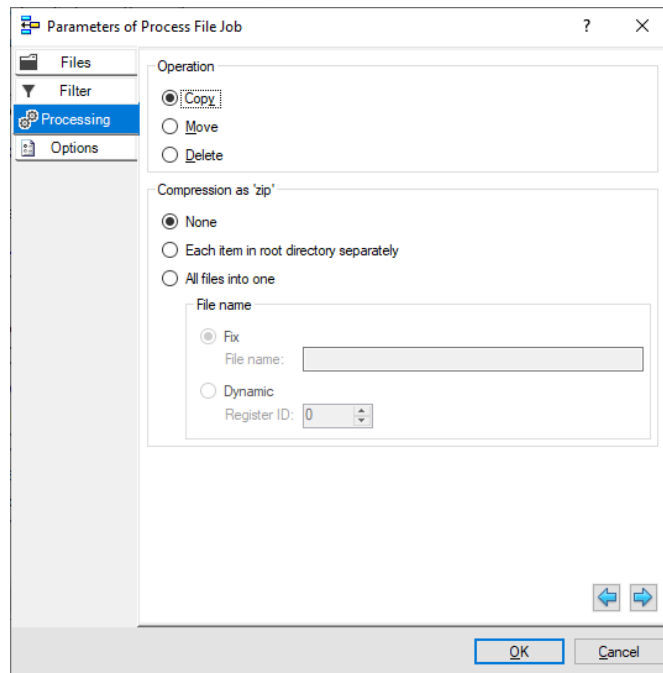


The **Filter** tab contains the following elements:

Element	Description
Include specific files	Allow only files with a specific name. You can provide multiple names that have to be separated by a semicolon. Use * as wildcard
Fix	Fix filter.
Dynamic	Filter is dynamically read from a string register.
Exclude specific files	Exclude files with a specific name. You can provide multiple names that have to be separated by a semicolon. Use * as wildcard
Include specific folders	Allow only folders with a specific name. You can provide multiple names that have to be separated by a semicolon. Use * as wildcard
Process files and folders older than	Check if the number of files and folders to process should be limited by their last modified date.
Timespan	Only those items which are older than the actual time minus the timespan will be processed.
Process oldest files only and ignore newest	Check if you want to process the oldest and ignore the newest files.
Number of files to ignore	Specify the number of most recent files by their last modified date that should not be processed.

## Process File Job: Parameter Dialog Processing

▾ Screenshot of Parameter Dialog

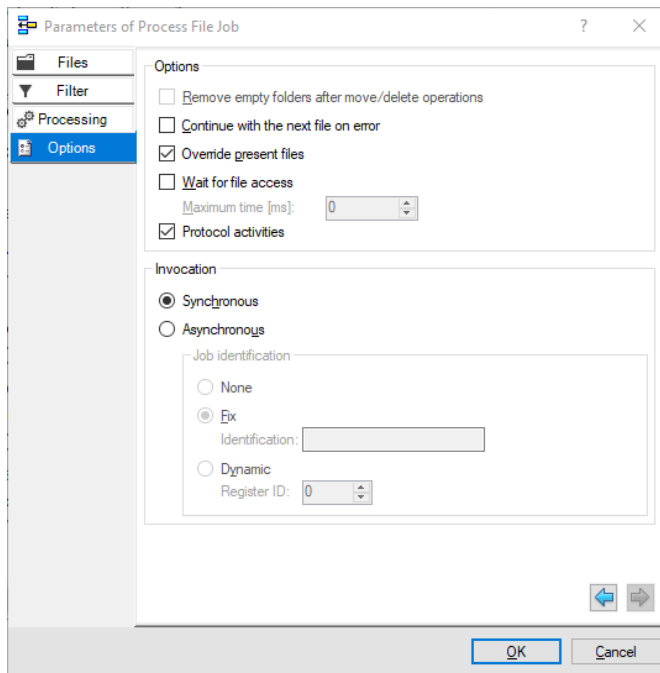


The **Processing** tab contains the following elements:

Element	Description
Copy	Check if you want to copy the files/folders to the target.
Move	Check if you want to move the files/folders to the target.
Delete	Check if you want to delete the files/folders in the source directory.
Compression as 'zip'	Specify if you want to use compression or not.
None	No compression. Files will be moved/copied as they are.
Each item in root directory separately	Check if you want to create a zip file from each item in the root directory. Every file will be put into its own zip file and each folder will get its own zip file. The names of these zip files are the file/folder names with a .zip at the end.
All files into one	All processed files and folders are put into one zip file. You have to provide the name for it.
Fix	Fix zip file name.
Dynamic	Dynamic zip file name from a string register.

## Process File Job: Parameter Dialog Options

▾ Screenshot of Parameter Dialog



The **Options** tab contains the following elements:

Element	Description
Remove empty folders after move/delete operations	Check if empty folders that are left after move/delete operations should be searched and deleted in the source directory.
Override present files	Check if files with the same name that are already in the target directory should be overwritten. Otherwise, an error will occur.
Wait for file access	Check if there should be a waiting time if a file can not be accessed, for example because it is locked by an application.
Maximum time [ms]	Waiting time for each file to be accessed.
Protocol activities	<p>Check if activities like copying, deleting etc. should be written into a protocol. It can be accessed from the menu <code>Tools -&gt; PI_FileManagement log viewer</code>. Each start of a check routine with an enabled protocol is in there.</p> <p>The entries can be identified by the time the check function was started. If you renamed your "Process File Job" check function, the new name will be in the name of the entry, too. The job identifier, if you used one, is also there.</p>
Synchronous	Select this if the check function should wait till all the files and folders are processed. If the files are big or if you have a slow network or hard drive then it can take some time to finish.
Asynchronous	Select this if the check function should start the processing in the background and finish. The check result will be OK if the processing could be started without a problem.
Job identification	Specify if and how a job should be uniquely named. The here defined name of the Job has to be used as parameter in the check function "Wait for File Job" in order to be able to wait for this job at another point of the check routine.
None	No identification for the job. If you start it, you can't check if it finished or not.
Fix	Fix job identifier.
Dynamic	Dynamic job identifier from a string register.

## Wait For File Job: Introduction

### Function overview

This check function can be used as synchronization point between an asynchronous running File Job and a check routine. If you insert this check function, the check routine will wait at this point, until a previous started asynchronous File Job has finished. You only have to specify the identification of the Job you are waiting for. It is also possible to set a timeout if you e.g. just want to check if the job is finished or not. If the job is finished, the check routine result will be OK, otherwise it will be NOK.

### Input data

This check function requires no input data object.

### Output data

The check function has no output data object.


### Result view

The check function has no result view.

In case of an error, the error message is displayed in the result view.

### Properties

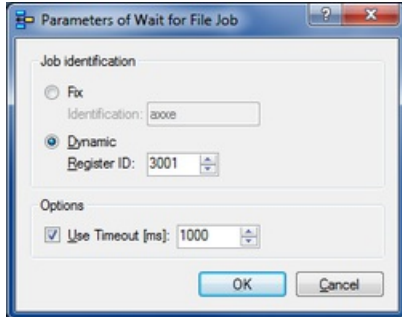
 Check function group Plug-In.

 The check function has a [Parameter Dialog](#).

## Wait For File Job: Parameter Dialog

This plug-in check function has a **Parameter** dialog.

### ☑ Screenshot of Parameter Dialog



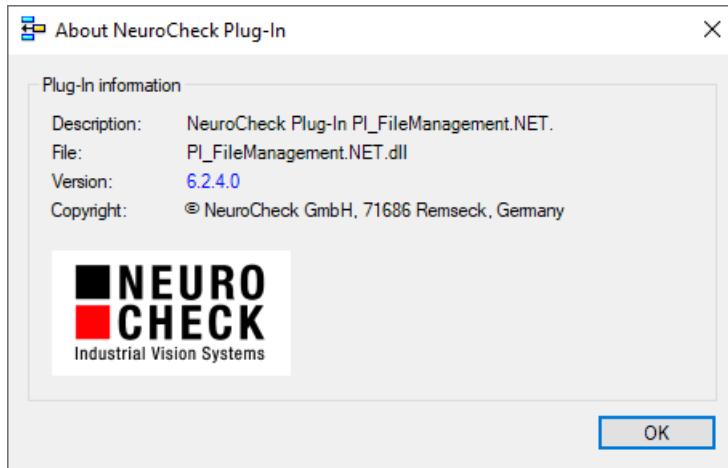
The **Parameter** dialog contains the following elements:

Element	Description
Job identification	Specify if you want to use a fix or dynamic job identifier. It is important to use here the identical identification you have defined in the check function "Process File Job" (in the tab page "Invocation"), which has started that job asynchronous you are waiting for.
Fix	Fix job identifier.
Dynamic	Dynamic job identifier from a string register.
Use Timeout [ms]	Check if a timeout should be used that the check function will wait for the job to complete.

## About Dialog

This dialog displays version information about the NeuroCheck Plug-In **PI\_FileManagement.NET.dll**.

[Screenshot of About Dialog](#)



## Support Services

For technical support, please contact your local NeuroCheck partner or NeuroCheck GmbH:

Phone: +49 (0) 7146 - 89 56-40

E-Mail: [support@neurocheck.com](mailto:support@neurocheck.com)

Web: [www.neurocheck.com](http://www.neurocheck.com)

Before contacting us, please provide some important information about your system:

Information about your NeuroCheck installation and your PC setup:

- Use the NeuroCheck Diagnostics tool to check your installation and computer configuration.
- The NeuroCheck Diagnostics is installed in the "Tools" folder within your NeuroCheck installation.

Log file information:

- Logging for NeuroCheck can be activated in **System > Software Settings > Diagnosis > Logging**.

