

NXPowerLite™

File Server Edition

Neuxpower
 Studio 400 Highgate Studios
 London NW5 1TL
 United Kingdom

T: +44 (0)20 7424 8700
 F: +44 (0)20 7424 8701
 E: info@neuxpower.com
 W: www.neuxpower.com

User Manual

File Server Edition, Version 3.7

Copyright © 2007-2008 Neuxpower Solutions Ltd

1	Introduction.....	3
1.1	Overview	3
1.2	Important Information.....	3
1.3	Upgrading from NXPowerLite Server Edition 3.5 or Earlier	3
1.4	Upgrading from NXPowerLite Batch Process.....	3
2	Evaluating NXPowerLite File Server Edition	5
2.1	Install.....	5
2.2	Remote Desktop Configurations	5
2.3	Evaluation Wizard.....	5
2.4	Manual Evaluation.....	5
2.4.1	Configure a Dry Run.....	5
2.4.2	Start the Dry Run	6
2.4.3	Run in “Optimize” Mode	6
3	Deploying NXPowerLite File Server Edition in a Production Environment	8
3.1	Overview	8
3.2	Configure COM+ Security for NXPowerLite File Server Edition.....	8
3.2.1	Identities.....	8
3.2.2	COM+ Security Configuration	8
3.3	Configure NXPowerLite File Server Edition	9
3.4	Launch NXPowerLite File Server Edition.....	9
4	Command-Line Options	11
5	Configuration.....	12
5.1	Overview	12
5.2	Properties	12
5.2.1	Boolean Properties	12
5.2.2	Number Properties	12
5.2.3	String Properties.....	12
5.2.4	Array Properties.....	13
5.3	Rules.....	13
5.4	Batch Properties Reference	13
5.4.1	DebugLevel	13
5.4.2	FileExtensions	14
5.4.3	MaxRunTime.....	14
5.4.4	MinAge	14
5.4.5	MinSize	14
5.4.6	MinSizeDelta.....	14
5.4.7	OptimizeOfflineFiles	15
5.4.8	OptimizeReadOnlyFiles.....	15
5.4.9	RunMode	15
5.4.10	SearchPaths	16
5.4.11	SearchPathsExclude.....	16
5.5	Optimization Properties Reference.....	16

NXPowerLite™

File Server Edition

Neuxpower
Studio 400 Highgate Studios
London NW5 1TL
United Kingdom

T: +44 (0)20 7424 8700
F: +44 (0)20 7424 8701
E: info@neuxpower.com
W: www.neuxpower.com

5.5.1	AllowCropping	16
5.5.2	AllowJpeg	16
5.5.3	AllowResizing	17
5.5.4	DotsPerInch	17
5.5.5	FlattenEmbeddedObjects	17
5.5.6	JpegQuality	17
5.5.7	PreserveFileTimes	18
5.5.8	PreserveSecurityDescriptor	18
5.5.9	ReplaceFilesByCopying	18
5.5.10	TargetHeight	18
5.5.11	TargetWidth	18
5.6	Rules Reference	19
5.6.1	Rules	19
5.6.2	Conditions	19
5.6.3	Actions	20
6	Sample Configuration File	21

1 Introduction

1.1 Overview

NXPowerLite File Server Edition is an application using NXPowerLite optimization technology to recursively search through a configurable set of directories, optimizing files and replacing the original files with optimized versions.

NXPowerLite File Server Edition is designed to run unattended; it has no user interface, and all configuration is done using one or more text-based configuration files.

1.2 Important Information

NXPowerLite File Server Edition automatically overwrites files with optimized versions of themselves, and the compression technology is “lossy” by design – excess information is stripped from files as part of the optimization process.

Depending on the specific circumstances, it is always possible that some users might be unhappy with the results of optimizing their files using centrally-determined optimization settings.

As a result, we ***strongly recommend*** that any files in directories that will be optimized by NXPowerLite are safely backed up before running the script in “optimize” mode.

1.3 Upgrading from NXPowerLite Server Edition 3.5 or Earlier

The product previously known as NXPowerLite Server Edition is now named NXPowerLite File Server Edition. Upgrading customers should be aware of the following:

- There is a new RunMode mandatory configuration property. To preserve the same behaviour as previous versions, you will need to add the following line to your configuration file:

```
config.RunMode = 'optimize';
```

Please see section 5.4.9 for more information about the RunMode property.

- When running interactively (i.e. without passing the batch mode “/b” option to wscript.exe) NXPowerLite will now display message boxes under various circumstances, as well as writing events to the event log. You should ensure that wscript.exe is launched with “/b” command-line option in your production environment, otherwise the script will not work properly.

1.4 Upgrading from NXPowerLite Batch Process

NXPowerLite File Server Edition replaces the previously available “NXPowerLite Batch Process”. Upgrading customers should be aware of the following:

- Before installing NXPowerLite File Server Edition, please ensure that a backup copy has been made of the configuration file (**config.js**). The uninstaller for NXPowerLite Batch Process will delete this file.

NXPowerLite™

File Server Edition

Neuxpower
Studio 400 Highgate Studios
London NW5 1TL
United Kingdom

T: +44 (0)20 7424 8700
F: +44 (0)20 7424 8701
E: info@neuxpower.com
W: www.neuxpower.com

- The name of the primary configuration file is now **nxplbatch.cfg** rather than **config.js**. However, the format of this file is backwardly compatible, so the existing configuration file can simply be renamed.
- The changes in the section above (Upgrading from NXPowerLite Server Edition 3.5 or Earlier) also apply.

2 Evaluating NXPowerLite File Server Edition

2.1 Install

NXPowerLite File Server Edition is provided as a standard Windows Installer .msi package. For evaluation purposes, installation is as simple as double-clicking the .msi file and following the Setup Wizard instructions.

If you are deploying NXPowerLite into a production environment, please see section 3 (Deploying NXPowerLite File Server Edition in a Production Environment) instead.

2.2 Remote Desktop Configurations

The following instructions assume that the user is logged in interactively to the server. If you are evaluating NXPowerLite File Server Edition via a Remote Desktop connection, some additional configuration is required.

Please see section 3.2 (Configure COM+ Security for NXPowerLite File Server Edition) for details.

2.3 Evaluation Wizard

For your convenience, NXPowerLite File Server Edition includes an Evaluation Wizard. This Wizard will guide you through the evaluation process, enabling you to easily perform a “dry run” with just a few mouse clicks. The dry run process will optimize your files without overwriting the original files, and generate a report showing various statistics including the amount of space saved. The optimized copies of your files will then be automatically deleted. Your original files will not be modified in any way. To perform a dry run, simply open the Evaluation Wizard and follow the on-screen instructions. A shortcut to the Evaluation Wizard can be found under Start Menu / All Programs / NXPowerLite File Server Edition.

2.4 Manual Evaluation

If you prefer, you can perform a manual evaluation of NXPowerLite File Server Edition instead of using the Evaluation Wizard.

2.4.1 Configure a Dry Run

- In Windows Explorer, open the NXPowerLite File Server Edition installation directory (by default, this is **C:\Program Files\Neuxpower Solutions Ltd\NXPowerLite File Server Edition**, or **C:\Program Files (x86)\Neuxpower Solutions Ltd\NXPowerLite File Server Edition** on 64-bit systems).
- Make a copy of the sample configuration file **nxplbatch.cfg.eval** and rename the copy to **nxplbatch.cfg**. If you're upgrading from a previous version, you may already have a file named **nxplbatch.cfg** in this directory.
- Open the file **nxplbatch.cfg** in your preferred text editor.

- Locate the SearchPaths property and set it to point to your directory of test files. For example, if your test files are in the directory **C:\testfiles**, change the SearchPaths line to read as follows, making sure any backslash characters are doubled as shown:

```
config.SearchPaths = ['C:\\testfiles'];
```

- Locate the FileExtensions property and make sure it includes all the file types that appear in your test directory. The sample configuration includes all Office 97-2007 file formats supported by NXPowerLite.

```
config.FileExtensions = ['pps', 'ppt', 'pot', 'doc', 'dot', 'xls', 'xlt', 'potm', 'potx', 'ppsm', 'ppsx', 'pptm', 'pptx', 'docm', 'docx', 'dotm', 'dotx', 'xlsm', 'xlsx', 'xlsm', 'xltx'];
```

- Save and close the configuration file.

2.4.2 Start the Dry Run

To start NXPowerLite, double-click the file **nxplbatch.js** in the NXPowerLite File Server Edition installation directory.

If there is a problem with the configuration, a message box will pop up at this point showing one or more error messages. If this occurs, please go through the steps in section 2.4.1 (Configure a Dry Run) again and check that everything is as it should be. The error message(s) should give some indication as to the nature of the problem.

If the configuration is OK, a message box will pop up asking for permission to continue. After you answer “Yes”, NXPowerLite will not open any windows or generate any output until it has finished. To see what it is doing, you can open the Windows Event Viewer and look at the Application Event Log. Events from NXPowerLite File Server Edition will have their Source listed as “WSH”.

The Event Log should contain the following:

- An Information event with the message “NXPowerLite batch process starting”
- A number of Information events stating “About to optimize [filename]”, each of which should be followed either by another Information event stating “Optimized [filename] [original size] => [optimized size]” or a Warning event indicating that a problem occurred optimizing the file.

When NXPowerLite has finished, another event will be logged beginning with “NXPOWERLITE FILE SERVER EDITION REPORT”. This will be followed by a report of the results, broken down by file type.

2.4.3 Run in “Optimize” Mode

It's also possible to evaluate NXPowerLite File Server Edition in “optimize mode”. In this mode, NXPowerLite will optimize files that match the configured criteria and overwrite original files with optimized versions.

To run NXPowerLite File Server Edition in “optimize mode”, open the file “nxplbatch.cfg” in your preferred text editor, locate the RunMode property and change it from “dryrun” to “optimize” as shown:

```
config.RunMode = 'optimize';
```

NXPowerLite™

File Server Edition

Neuxpower
Studio 400 Highgate Studios
London NW5 1TL
United Kingdom

T: +44 (0)20 7424 8700
F: +44 (0)20 7424 8701
E: info@neuxpower.com
W: www.neuxpower.com

Please note that, as per the terms of the Evaluation Agreement, you are not permitted to evaluate NXPowerLite File Server Edition in “optimize mode” in a production environment. You may only evaluate in this mode on a non-production machine, using a selection of test files. Copies of the Evaluation Agreement are available, upon request, from support@nxpowerlite.com.

3 Deploying NXPowerLite File Server Edition in a Production Environment

3.1 Overview

The setup package installs the NXPowerLite optimizer component into a COM+ application. By default, this application is set to run as the interactive user (i.e. the user who is currently logged in at the server console) and access checks are disabled. This may be acceptable for testing purposes, but in order to have the application work properly and securely in a production environment, it is necessary to enable access checks and set the identity under which the application will run.

PLEASE NOTE: The following process must be repeated after updating to a newer version of NXPowerLite File Server Edition. The update process will reset all security-related properties of the COM+ application to their default values.

3.2 Configure COM+ Security for NXPowerLite File Server Edition

3.2.1 Identities

There are two identities associated with NXPowerLite:

- The “script identity”. This is the user account under which the nxplbatch.js script will be launched.
- The “component identity”. This is the user account under which the NXPowerLite File Server Edition COM+ Application will run.

The simplest approach is to use the same user account for both of these purposes. The account will need to have full control over files in the directories to be optimized, and will also require the “Manage auditing and security log” privilege for the PreserveSecurityDescriptor feature to be able to preserve audit policy information for optimized files. Depending on your security configuration, an Administrator account is likely to satisfy these requirements.

For additional rights and permissions required for COM+ application identity accounts, please see this Microsoft knowledge base article:

<http://support.microsoft.com/default.aspx/kb/276407>

3.2.2 COM+ Security Configuration

- First, start the Component Services tool. This can be found in one of the following locations:

Operating System	Component Services tool location
Windows 2000	Start / Settings / Control Panel / Administrative Tools
Windows XP	Start / Control Panel / Administrative Tools OR Start / Control Panel / Performance and Maintenance / Administrative Tools
Windows Server 2003	Start / Administrative Tools OR Start / Settings / Control Panel / Administrative Tools

- In the Component Services tool, navigate to Computers / My Computer / COM+ Applications.

- Right-click on the “NXPowerLite File Server Edition” application icon and select “Properties”.
- Select the “Security” tab.
- Check the “Enforce access checks for this application” checkbox.
- Select the “Identity” tab.
- Click the “This user:” radio button and select the “component identity” account (see section 3.2.1 for details).
- Click the “OK” button to save your changes.
- Under the “NXPowerLite File Server Edition” application, expand Roles / OptimizerUser / Users.
- Right-click on the “Users” folder and select New / User from the menu.
- Select the “script identity” account (see section 3.2.1 for details) and click “OK”.

3.3 Configure NXPowerLite File Server Edition

Before using NXPowerLite File Server Edition, it must be configured. Configuration allows you to specify:

- Whether to actually optimize and replace files, or just to generate a report without modifying any files. See section 5.4.9 (RunMode) for more information.
- The criteria to be used in selecting files to be optimized. See section 5.4 (Batch Properties Reference) for more information.
- The settings to use when optimizing files. See section 5.4.11 (SearchPathsExclude

Type	Array
Default Value	[] (empty)

SearchPathsExclude is a list of Windows directory paths that should be excluded from optimization. Any directories that appear in this list will not be searched for files to optimize, and neither will any of their subdirectories. See the example below:

```
config.SearchPathsExclude = ['\\\\server\\share\\marketing'];
```

Please see section 5.4.10 (SearchPaths) above for more information about path separator characters (\ and /).

- Optimization Properties Reference) for more information.
- “Rules” which allow different settings to be used for different files, based on various criteria. See section 5.6 (Rules Reference) for more information.

The setup package places a sample configuration file (**nxplbatch.cfg.sample**) in the installation directory (default: **C:\Program Files\Neuxpower Solutions Ltd\NXPowerLite File Server Edition**). This configuration file can be used as a starting point; simply rename the file to **nxplbatch.cfg** once it has been customized for your requirements.

The sample configuration file contains examples of every available configuration property, along with comments briefly describing them.

3.4 Launch NXPowerLite File Server Edition

Before launching NXPowerLite File Server Edition in “optimize” mode, it is *very important* to ensure that backup copies have been made of any files in directories that will be optimized – that is, any directories that are referred to in the SearchPaths property, and any descendants of those directories. Please see section 1.2 (Important Information) for further details.

Before NXPowerLite is run unattended (e.g. as part of a scheduled task) please check the following:

- **nxplbatch.js** *must* be started with the “/b” (batch mode) option supplied to Windows Script Host, otherwise it will attempt to interact with the user via message boxes. The following command will start NXPowerLite with Windows Script Host in batch mode:

```
wscript /b \path\to\nxplbatch.js
```

- Make sure COM+ application security has been configured as described in section 3.2 (Configure COM+ Security for NXPowerLite File Server Edition) and that **nxplbatch.js** is to be launched under the “script identity” account.

4 Command-Line Options

The `nxplbatch.js` script accepts the following command-line options:

<code>/nodefaultconfig</code> <code>/n</code>	Prevents the primary configuration file (<code>nxplbatch.cfg</code>) from being read. This can be useful, for example, if you wish to keep your configuration file(s) somewhere other than in the NXPowerLite File Server Edition installation directory.
<code>/config <filename></code> <code>/c <filename></code>	Configuration will be read from the specified file. This option can appear more than once, causing NXPowerLite to read all configuration files in the order they are provided on the command line. It is possible to use this feature to override specific properties.

For example, the following command will look for configuration in the file

`c:\configs\nxplbatch.cfg` instead of the default location (`nxplbatch.cfg` in the installation directory):

```
wscript /b \path\to\nxplbatch.js /nodefaultconfig /config c:\configs\nxplbatch.cfg
```

Alternatively, it is possible to use multiple configuration files. For example, if NXPowerLite File Server Edition is running on a nightly basis, but there is a requirement to run the script against a different set of directories depending on the day of the week. On Monday, this command would execute (using the short form of the command-line options):

```
wscript /b \path\to\nxplbatch.js /n /c c:\configs\default.cfg /c c:\configs\monday.cfg
```

A similar command would run on Tuesday, but specifying `tuesday.cfg` instead of `monday.cfg` as the second configuration file.

In this situation, `default.cfg` would contain configuration properties applicable to every day, and the daily configuration files would override only the properties that needed to differ from the defaults in `default.cfg`.

For example, `monday.cfg` could be simply:

```
config.SearchPaths = ['//server1/docs'];
```

and `tuesday.cfg` could be:

```
config.SearchPaths = ['//server2/clients'];
```

In this situation, the `SearchPaths` property in each of the secondary configuration files would override the `SearchPaths` property in the primary configuration file.

5 Configuration

5.1 Overview

The main NXPowerLite File Server Edition configuration file is named **nxplbatch.cfg** and should be placed in the installation directory (default: **C:\Program Files\Neuxpower Solutions Ltd\NXPowerLite File Server Edition**).

NXPowerLite Server File Edition can also read alternative or additional configuration files; see section 4 (Command-Line Options) for further details.

5.2 Properties

Basic configuration is achieved through the use of **properties**. Property definitions take the following form:

```
config.PropertyName = value;
```

Currently, there are three types of properties: Boolean, Number and Array.

5.2.1 Boolean Properties

Boolean values can be either *true* or *false*, as in the following examples:

```
config.AllowCropping = true;  
config.FlattenEmbeddedObjects = false;
```

5.2.2 Number Properties

The permissible values for Number properties are specified in the documentation for individual properties. The following is an example of a Number property:

```
config.TargetWidth = 1024;
```

Some Number properties allow units to be specified, as in the following examples:

```
config.MinAge = weeks(4);  
config.MinSize = megabytes(1);
```

The available time-related units are: seconds, minutes, hours, days, weeks.

The available size-related units are: bytes, kilobytes, megabytes.

5.2.3 String Properties

The permissible values for String properties are specified in the documentation for individual properties. The following are examples of String properties; please note that strings may be surrounded either by double quotes (") or single quotes ('):

```
config.RunMode = 'dryrun';  
config.RunMode = "optimize";
```

5.2.4 Array Properties

Array properties can contain more than one value. The following are examples of array properties:

```
config.SearchPaths = ['c:/documents', '//server/share'];  
config.FileExtensions = ['ppt', 'ppt', 'pot'];
```

Please see the documentation for the SearchPaths property (section 5.4.10) for more information about the use of slash (“/” and “\”) characters in path names.

5.3 Rules

NXPowerLite File Server Edition supports the use of **rules** to provide greater control over the optimization settings that are applied.

It may be appropriate to use rules if either of the following apply:

- The criteria deciding whether or not to optimize files are more complex than those offered by the MinAge, MinSize and MinSizeDelta properties.
- There is a requirement to divide files into groups based on their properties, and apply different optimization settings to each group.

Please see section 5.6 (Rules Reference) for more information about rules.

5.4 Batch Properties Reference

5.4.1 DebugLevel

Type	Number
Default Value	0

Because NXPowerLite File Server Edition is intended to be run as an unattended process, any output is sent to the Application Event Log. The DebugLevel property determines the amount of information that will be written to the Event Log, as follows:

0. **Normal:** A single event is logged when the process begins, and another (giving a summary of the results achieved) is logged when the process ends.
1. **Show Errors:** In addition to the events from level 0, warning events are logged when an error occurs optimizing a file.
2. **Chatty:** In addition to the events from levels 0 and 1, informational events are logged after optimizing each file.
3. **Debug:** In addition to the events from levels 0 to 2, informational events are logged before optimizing each file. This can be useful in troubleshooting situations where a particular file takes an unusually long time to optimize, for example.

In addition to the events described above, NXPowerLite will always log error events if the configuration file is not valid for some reason.

5.4.2 FileExtensions

Type	Array
Default Value	['pps','ppt','pot']

FileExtensions is a list of file extensions to optimize. These must be taken from the set of supported PowerPoint, Word or Excel file extensions, which are as follows:

PowerPoint	pot, pps, ppt , pptx, pptm, potx, potm, ppam, ppsx, ppsm
Word	doc, dot, docx, docm, dotx, dotm
Excel	xls, xlt, xlsx, xlsx, xltm, xltm, xlam

5.4.3 MaxRunTime

Type	Number
Default Value	hours(4)

MaxRunTime is the maximum time allowed for the process to run. This is checked after each file is optimized, so actual runtime may exceed this value by several minutes.

5.4.4 MinAge

Type	Number
Default Value	0

MinAge is the minimum time that must have elapsed since a file was last modified for it to be optimized. For example, if this property is set to weeks(8), only files that have not been modified in the last 8 weeks will be optimized. The default setting (0) means that files will be optimized regardless of their last-modified date.

5.4.5 MinSize

Type	Number
Default Value	megabytes(1)

MinSize is the minimum size of files to optimize. Any files smaller than this threshold will not be optimized.

5.4.6 MinSizeDelta

Type	Number
Default Value	megabytes(1)

MinSizeDelta is the minimum size increase that must have occurred since a file was last optimized for it to be optimized again.

For example, if this property is set to megabytes(1), the default, only files that have increased in size by at least 1 megabyte since they were last optimized will be reoptimized.

This property only takes effect for files that have previously been optimized with NXPowerLite 3.0.5 or later; earlier versions of NXPowerLite did not “tag” optimized files with the necessary information for this feature to be enabled. Such files will be optimized on the first run, but will be “tagged” to allow them to be properly handled in the future.

5.4.7 OptimizeOfflineFiles

Type	Boolean
Default Value	False

OptimizeOfflineFiles determines whether or not files with the “offline” attribute will be optimized. Generally, this should be set to *false* (the default) to avoid the performance overhead of retrieving files from offline storage (e.g. a tape library) in order to optimize them.

5.4.8 OptimizeReadOnlyFiles

Type	Boolean
Default Value	False

OptimizeReadOnlyFiles determines whether or not files with the “read-only” attribute will be optimized. If this property is *false* (the default), read-only files will not be optimized. If the value is *true*, read-only files will be optimized and returned to read-only status after optimization.

5.4.9 RunMode

Type	String
Default Value	'dryrun'

NXPowerLite File Server Edition can run in 3 possible modes. Possible values of this property are:

'optimize'	Normal operation mode. NXPowerLite File Server Edition will optimize files that match the configured criteria and overwrite original files with optimized versions. When the process completes, an Event Log entry will be written containing a report showing file sizes before and after optimization.
'dryrun'	“Dry run” mode. NXPowerLite File Server Edition will optimize files as above, but will not overwrite original files or modify them in any way. Optimized files are created in a temporary location, then immediately deleted once their size has been noted. The report generated at the end of the process is identical to the report generated in “optimize” mode. This mode can be used to test the compression achievable with a particular configuration.
'stats'	In this mode, NXPowerLite File Server Edition will not optimize files at all. This means that the process will complete far more quickly than the other two modes. The report generated at the end will contain everything except details of optimized file sizes and space saved. This mode can be used to quickly determine the number and total size of files matching the configured criteria.

5.4.10 SearchPaths

Type	Array
Default Value	[] (empty)

SearchPaths is a list of Windows directory paths containing files to be optimized. Each directory in turn and all its subdirectories will be searched for files to optimize. See the example below:

```
config.SearchPaths = ['c:/documents', '//server/share'];
```

Please note the use of “forward slash” characters (/) rather than backslashes (\) in path names. This is because the backslash (\) is used as an “escape” character. It is acceptable either to use forward slashes as path separators, as in the example above, or to use backslashes and ensure they are always doubled, as in the example below:

```
config.SearchPaths = ['c:\\documents', '\\\\server\\share'];
```

5.4.11 SearchPathsExclude

Type	Array
Default Value	[] (empty)

SearchPathsExclude is a list of Windows directory paths that should be excluded from optimization. Any directories that appear in this list will not be searched for files to optimize, and neither will any of their subdirectories. See the example below:

```
config.SearchPathsExclude = ['\\\\server\\share\\marketing'];
```

Please see section 5.4.10 (SearchPaths) above for more information about path separator characters (\ and /).

5.5 Optimization Properties Reference

5.5.1 AllowCropping

Type	Boolean
Default Value	true
File Types	All

AllowCropping determines whether images will be cropped. If this property is *true*, any portions of an image that are not visible in the document due to cropping will be removed.

5.5.2 AllowJpeg

Type	Boolean
Default Value	true
File Types	All

AllowJpeg determines whether or not images will be compressed using the JPEG format, where appropriate. If this property is *true*, NXPowerLite will decide whether or not to use JPEG compression

on a per-image basis. If it is *false*, NXPowerLite will never use JPEG compression, although if the input file contains JPEGs these may still be present in the output file.

See also: JpegQuality

5.5.3 AllowResizing

Type	Boolean
Default Value	true
File Types	All

AllowResizing determines whether or not images will be physically resized according to the TargetWidth and TargetHeight properties. If this property is *true*, any oversized images will be scaled down. If it is *false*, images will never be resized.

See also: TargetHeight, TargetWidth

5.5.4 DotsPerInch

Type	Number
Default Value	75
File Types	Word, Excel

DotsPerInch sets the resolution in DPI of the target device. This is used in deciding the target dimensions for images in Word and Excel files when scaling them down (if AllowResizing is *true*).

See also: AllowResizing

5.5.5 FlattenEmbeddedObjects

Type	Boolean
Default Value	true
File Types	All

FlattenEmbeddedObjects determines if embedded OLE objects will be converted to images. If this property is *true*, embedded objects will always be converted where possible. If it is *false*, they will never be converted.

5.5.6 JpegQuality

Type	Number
Default Value	7
File Types	All

JpegQuality sets the quality to be used when compressing images using the JPEG format. This is a number from 1-9 inclusive, where 1 is the lowest quality and 9 is the highest. We have found 6 to offer a good compromise between size and quality, although results may vary according to the content of documents.

See also: AllowJpeg

5.5.7 PreserveFileTimes

Type	Boolean
Default Value	false
File Types	All

If this property is *true*, each file's creation, last modified and last accessed times will be preserved during the optimization process.

5.5.8 PreserveSecurityDescriptor

Type	Boolean
Default Value	false
File Types	All

If this property is *true*, NXPowerLite will preserve as much as possible of the security descriptor from each file optimized. If the process has sufficient access rights and privileges, this will preserve the file's owner, group, discretionary access control list (DACL) and system access control list (SACL).

5.5.9 ReplaceFilesByCopying

Type	Boolean
Default Value	False
File Types	All

If this property is *true*, an alternative mechanism is used to replace files.

This property should *only* be set to *true* if the normal mechanism causes problems (e.g. some customers with NetWare®-based file servers have reported that this setting is necessary in order for file ownership information to be preserved).

5.5.10 TargetHeight

Type	Number
Default Value	960
File Types	PowerPoint

TargetHeight sets the height in pixels of the target display size for presentations. This is used in deciding the target height of images when scaling them down, so it is only relevant if the AllowResizing property is *true*).

See also: AllowResizing, TargetWidth

5.5.11 TargetWidth

Type	Number
Default Value	1280
File Types	PowerPoint

TargetWidth sets the width in pixels of the target display size for presentations. This is used in deciding the target width of images when scaling them down, so it is only relevant if the AllowResizing property is *true*).

See also: AllowResizing, TargetHeight

5.6 Rules Reference

Rules are only tested for files that satisfy the basic criteria specified by configuration properties (i.e. MinAge, MinSize and MinSizeDelta).

A set of rules begins with “BEGIN RULES” and ends with “END RULES”, each on a line by itself. So an empty rule set would be just:

```
BEGIN RULES  
END RULES
```

Rules are tested in the order that they are specified.

If none of the rules matches a file, that file will be optimized using the settings specified by the optimization properties (“standard settings”).

5.6.1 Rules

An example rule is shown below:

```
rule:      Dormant  
condition: lastModified.isOlderThan(weeks(26))  
action:    optimize({TargetWidth:800, TargetHeight:600})
```

Each rule has an optional **name** (“Dormant” in the example above), zero or more **conditions** and exactly one **action**. The rule in this example will match files that were last modified more than 6 months ago and optimize them with the TargetWidth and TargetHeight properties set to 800 and 600 respectively, but all other properties as in the standard settings.

Rule names are used for two purposes:

1. To provide error feedback (i.e. if a rule fails to compile, the error message in the Application Event Log will include the rule name)
2. In the summary that is logged at the end of the optimization process

5.6.2 Conditions

Zero or more conditions may be specified for each rule; the rule is said to “match” a file if **all** of its conditions are satisfied. If no conditions are specified, the rule matches all files.

Conditions are specified using JavaScript syntax. Currently available properties are:

Property name	Description
path	File’s full directory path.
lastModified	Date/time when the file was last modified.
created	Date/time when the file was created.
size	Size of the file, in bytes.
lastOptimizedSize	Size of the file after it was last optimized, in bytes. This will be zero if the file has never been optimized.

The operations currently supported on dates (i.e. the *lastModified* and *created* properties) are *isOlderThan()* and *isNewerThan()*. For example:

```
condition: lastModified.isOlderThan(weeks(26))
```

matches files which have not been modified in the last 26 weeks.

Sizes can be compared using normal JavaScript comparison operators. For example:

```
condition: size > megabytes(2)
```

matches files which are larger than 2 megabytes in size. A more complex example:

```
condition: (lastOptimizedSize == 0) || (size - lastOptimizedSize > megabytes(1))
```

matches files which have never been optimized, or whose size has increased by more than 1 megabyte since they were last optimized.

5.6.3 Actions

Each rule must have exactly one action, which is performed if the current file matches the rule.

Available actions are:

Action	Description
skip()	Causes the file to be passed over; it will not be optimized
optimize(settings)	Causes the file to be optimized with the specified settings. Only settings that differ from standard settings need to be specified – see examples below.

For example, this action will cause matching files to be optimized with standard settings:

```
action: optimize({})
```

This action will optimize matching files with standard settings, except that JpegQuality is set to 9 and cropping is disabled:

```
action: optimize({JpegQuality:9,AllowCropping:false})
```

Please see section 6 (Sample Configuration File) for a complete example.

6 Sample Configuration File

```
// Configuration file for nxplbatch.js

// RunMode - Selects the mode of operation.
// Possible modes are:
// 'optimize' - normal operation; files will be replaced with optimized versions
// 'dryrun' - files will be optimized to find out how much they would have been reduced, but the
//           optimized versions will not replace the original files
// 'stats' - no optimization will be performed; a report will be generated showing the sizes of
//           files that would have been optimized, broken down in various ways
config.RunMode = 'dryrun';

// SearchPaths - List of Windows directory paths containing files to be optimized.
// Each directory in turn and all its subdirectories will be searched for files to optimize.
// Please note that backslashes must be doubled.
// EXAMPLE: config.SearchPaths = ["d:\\path\\to\\files", "e:\\path\\to\\files"];
config.SearchPaths = [];

// SearchPathsExclude - List of Windows directory paths to exclude from optimization.
// Directories appearing in this list and their subdirectories will not be searched.
// Please note that backslashes must be doubled.
// EXAMPLE: config.SearchPathsExclude = ["d:\\path\\to\\files\\excluded"];
config.SearchPathsExclude = [];

// MaxRunTime - Maximum time allowed for the process to run.
// We check whether this time has elapsed after each file is optimized, so actual runtime may in
// practice exceed this value by several minutes.
// Times may be specified in seconds, minutes or hours.
config.MaxRunTime = hours(4);

// FileExtensions - List of file extensions to optimize.
// These must be taken from the set of PowerPoint, Word or Excel extensions
// (i.e. DOC, DOT, PPT, PPS, POT, XLS or XLT).
config.FileExtensions = ['pps', 'ppt', 'pot', 'doc', 'dot', 'xls', 'xlt'];

// MinSize - Minimum size, in bytes, of files to optimize.
// Any files smaller than this threshold will not be processed.
// Sizes may be specified in bytes, kilobytes or megabytes.
config.MinSize = megabytes(1);

// MinSizeDelta - Minimum size increase, in bytes, that must have occurred since a file was last
//                optimized for it to be optimized again.
// For example, if this property is set to megabytes(1), as below, only files that have increased
// in size by at least 1MB since they were last optimized will be reoptimized. This setting has no
// effect on files that have not previously been optimized with NXPowerLite 3.0.5 or later.
config.MinSizeDelta = megabytes(1);

// MinAge - Minimum time, in seconds, that must have elapsed since a file was last modified for it
//          to be optimized.
// For example, if this property is set to weeks(8), only files that have NOT been modified in the
// last 8 weeks will be optimized. The default setting (0) means that files will be optimized
// regardless of their last-modified date.
config.MinAge = 0;
```

```
// DebugLevel - Verbosity level for debugging purposes.
// Possible values are:
// 0 - Normal level; a single event is logged when the process begins, and another when it ends.
// 1 - Show errors; in addition to the events from level 0, warning events are logged when an error
// occurs optimizing a file.
// 2 - Chatty; in addition to the events from levels 0 and 1, informational events are logged after
// optimizing each file.
// 3 - Debug; in addition to the events from levels 0-2, information events are logged before
// optimizing each file.
config.DebugLevel = 0;

// AllowCropping - Should images have cropped portions removed?
config.AllowCropping = true;

// AllowJpeg - Should we convert images to JPEG format where appropriate?
config.AllowJpeg = true;

// AllowResizing - Should images be scaled down?
config.AllowResizing = true;

// DotsPerInch - Sets the target resolution for optimizing Word and Excel files.
// This value is ignored unless AllowResizing is enabled.
config.DotsPerInch = 75;

// JpegQuality - Sets the quality level for JPEG compression.
// This value is ignored unless AllowJpeg is enabled.
config.JpegQuality = 7;

// PreserveFileTimes - Should optimized files have the same creation, last modification and last
// access dates/times as the original files?
config.PreserveFileTimes = true;

// PreserveSecurityDescriptor - Should we attempt to copy the security descriptor (owner, ACL,
// audit information etc.) from the original file to the optimized
// file?
// The process will copy as much security information as possible (including enabling the
// SeSecurityName privilege, if available, to copy the system ACL). However, the running user's
// access rights and privileges will limit the information that can be copied.
config.PreserveSecurityDescriptor = true;

// TargetHeight - Sets the target screen height in pixels for optimizing PowerPoint files.
// This value is ignored unless AllowResizing is enabled.
config.TargetHeight = 960;

// TargetWidth - Sets the target screen width in pixels for optimizing PowerPoint files.
// This value is ignored unless AllowResizing is enabled.
config.TargetWidth = 1280;

// FlattenEmbeddedObjects - Should we convert embedded OLE objects to pictures?
config.FlattenEmbeddedObjects = true;

// ReplaceFilesByCopying - Selects the method used for replacing files.
// This should be false unless there is a specific reason to do otherwise - see the NXPowerLite
// File Server Edition User Manual for details.
config.ReplaceFilesByCopying = false;

// OptimizeReadOnlyFiles - Should read-only files be optimized?
config.OptimizeReadOnlyFiles = false;

// OptimizeOfflineFiles - Should offline files be optimized?
config.OptimizeOfflineFiles = false;
```

```
// Rules - these allow optimization settings to be overridden based on complex criteria
// A set of rule specifications:
//   Begins with "BEGIN RULES" on a line by itself, and
//   Ends with "END RULES" on a line by itself
// Rules are checked in the order they are specified, on each file that satisfies the criteria
// specified above (e.g. MinAge, MinSize etc.).
//
// RULES
// =====
// For each rule:
// A NAME may optionally be given (e.g. "rule:Dormant" specifies that the following rule should be
// named "Dormant").
// Rule names are used to provide error feedback (i.e. if a rule fails to compile, the error
// message in the Application event log will include the rule name) and also in the summary that is
// logged at the end of the optimization process.
//
// Zero or more CONDITIONS (see below) may be specified; the rule is said to "match" a file if
// all of the conditions are satisfied. If no conditions are specified, the rule matches all files.
//
// Exactly one ACTION (see below) must be specified; if the rule matches a file, the action is
// performed, otherwise the next rule in sequence will be checked.
//
// If none of the rules match a file, the file will be optimized using the standard settings given
// above.
//
// CONDITIONS
// =====
// A rule can have zero or more conditions, which are specified using JavaScript syntax.
// Currently available properties are:
//   path           - file's full directory path
//   lastModified   - date when the file was last modified
//   created        - date when the file was created
//   size           - size of the file, in bytes
//   lastOptimizedSize - size of the file after it was last optimized, in bytes. This will be zero
//                   if the file has never been optimized
// For example, the first rule below ("condition:lastModified.isOlderThan(weeks(26))") matches if the
// file has not been modified in the last 26 weeks (approximately 6 months).
// The only operations currently supported on dates are isOlderThan() and isNewerThan().
// File size can be compared using normal JavaScript comparison operators, for example:
//   condition: size > megabytes(2)
// will match files larger than 2 megabytes.
//
// If no conditions are provided, the rule always matches.
// If more than one condition is provided, the rule matches only if ALL the conditions match.
//
// ACTIONS
// =====
// A rule must have exactly one action, which is performed if the current file matches the rule.
// Available actions are:
//   skip()         - causes the file to be passed over; it will not be optimized.
//   optimize(settings) - causes the file to be optimized with the specified settings. You need only
//                   specify settings that differ from the standard settings - see examples below.
// Examples:
// action:optimize({})
//   Optimizes the file with standard settings.
// action:optimize({JpegQuality:9,AllowCropping:false})
//   Optimizes with standard settings, except that JpegQuality is set to 9 and cropping is disabled.
//
// An example rule set follows.
BEGIN RULES
// Is file "dormant"? (i.e. hasn't been modified in the last 6 months)
// If so, optimize images for display at 800x600 pixels.
rule:      Dormant
condition: lastModified.isOlderThan(weeks(26))
action:    optimize({TargetWidth: 800, TargetHeight: 600})
```

NXPowerLite™

File Server Edition

Neuxpower
Studio 400 Highgate Studios
London NW5 1TL
United Kingdom

T: +44 (0)20 7424 8700
F: +44 (0)20 7424 8701
E: info@neuxpower.com
W: www.neuxpower.com

```
// Is file "semi-active"? (i.e. hasn't been modified in the last 2 months)
// If so, we don't want to flatten embedded objects.
rule:      Semi-Active
condition: lastModified.isOlderThan(weeks(8))
action:    optimize({FlattenEmbeddedObjects: false})
// If we get to this point, the file is "active" (i.e. has been modified in the last 2 months) so skip it
rule:      Active
action:    skip()
END RULES
```