Using the command line imposition tool in Quite Hot Imposing

Applies to version 6.0, March 2025

Quite Hot Imposing monitors hot folders and runs a command to impose each file. For advanced applications, developers might want to use the command line directly, bypassing the hot folder monitor.

These command line options can also be used:

- When using the hot folder monitor. Set under Advanced options for the queue.
- When using Enfocus Switch. Set as Other Options

Note that both the hot folder monitor and Enfocus Switch will automatically generate command line options, and you should avoid a clash. The -debugcmdline option can be used to display or log the command line as it is generated.

IMPORTANT NOTE. The hot folder monitor must always be run at least once, in order to set up licensing, even if the command line is always used thereafter.

Contents

Command line

The command is called **qi_applycommands.exe** on Windows, and **Quite Hot Engine** on macOS (the name is different in a demo version). In Windows it can be found in the same folder as the hot folder application. On macOS it is inside the **Contents/MacOS** folder of its own application bundle (which is not shown by Finder). See "default locations" below.

The parameters may appear in any order. Parameters are all identified by a keyword starting with a hyphen (-). Parameter values are separated by spaces. If any value contains a space, for example a file name, it should be enclosed in quotes according to the requirements of the operating system.

Required parameters

-control *filename* Name of an XML or PDF file containing imposition commands. filename can be just '-' to read XML (not PDF) from standard input. Alternatively, from version 4.0, **-nocontrol** can be used. This simply saves the source to the target. This is pointless unless combining files; by using **-nocontrol** the command line can be simply used as a tool to combine files.

-source *filename* Name of input PDF file. Starting with 4.0 this can also be the name of a directory. Unlike other parameters, **-source *filename*** can be repeated. See "Combining files" below for more detail.

-target *filename|dir* Target filename. If directory name is used alone, the filename is taken from the source[1]. This file will be overwritten if it exists. Starting in 5.0, certain commands may write multiple files. In this case **-target** is used only to choose a directory, and any filename part is ignored. See "Splitting files" below.

The input file and output file cannot be the same.

In macOS, file names are normal POSIX paths in UTF-8 format. For example "/Volumes/Macintosh HD/Work/PDFs/myinputfile.pdf".

Optional parameters

The **-bleed**, **-flatten**, **-pdfx** and **-registration** options each correspond to an option in the preferences of the hot folder process (and also in Quite Imposing Plus). Note that the command line process does *not* pick up preferences; if you want non-default values you must set them explicitly. These values can also be specified explicitly in the XML (in the user interface, by setting overriding preferences), which always take precedence.

-bleed Prompt	(default) If page to be imposed contains bleed, write a warning message (this utility does not actually prompt)
-bleed Use	If page to be imposed contains bleed, use it (use Trim Box.

[1] Note: before version 1.1 the form of -target with a directory did not work on macOS.

-bleed Ignore	If page to be imposed contains bleed, ignore it (use Crop Box.)
-flatten None	(default) Ignore form fields and annotations. If any appear on documents they will be lost in the imposition
-flatten FormsAndFields	Flatten printable form fields and annotations first.
-pdfx IgnoreAll	(default) Ignore PDF/X status of input file
-pdfx PreserveAll	Preserve PDF/X status of input file. Limits some functions. Refer to Quite Imposing Plus documentation for more details.
-registration AllSeps	(default) Registration marks are on all separations
-registration CMYK	Registration marks are 100% C,M,Y,K
-registration Black	Registration marks are black only
-userunit Allow	New in 6.0. Affects handling of large page sizes, known as “UserUnit” or “page scaling”. See manual under “Imposition Preferences” for discussion. With “Allow page scaling” page size allow for the page scaling. Page sizes larger than 200 x 200 inches (5040 x 5040 mm) can be created and set page scaling automatically. Some PDF applications will ignore the scaling and see a smaller page size.
-userunit Forbid	New in 6.0. With “Forbid”, then using a page with scaling set will give an error message, and the job will stop. Also, page sizes larger than 200 x 200 inches (5040 x 5040 mm) will not be created.
-userunit Warn	New in 6.0 With “Warn, a warning message is issued, and then work continues as if Allow is set.
-optimizeobjects	New in 4.0. When the result file is saved, check through the file for duplicate objects and consolidate them, which can make a smaller result file. This is not recommended as it may make saving files much slower, but is similar to processing done by Acrobat. This does not, however, optimize for “fast web view”.
-showversion	Add a line giving the product version
-fileinfo	Report info on the PDF such as size and PDF version

	before and after applying the commands
-successflag ***"value"***	New in 5.0. When Quite Hot Imposing has processed a file successfully, and written all result files, it will optionally write a success line to the standard error stream (stderr). This is intended for use by other apps wanting a robust way to check for success. Note that, according to platform, the line may be terminated by a carriage return (\r), a new line (\n), or both.
-fileflag ***"value"***	New in 5.0. Optionally write a line to the standard error stream (stderr) for each file that is written. In Quite Hot Imposing 5.0 and later, the Split/merge (partials) command may write multiple target files, and not write the target file at all. This is intended for use by other apps wanting a robust way to check output files. Note that, according to platform, this string will be followed immediately by the file name (no space), and the line may be terminated by a carriage return (\r), a new line (\n), or both.
-replacefile ***"filename"***	New in 6.0. Define a replacement file. All files with the same filename part are overridden from the control file. (See manual under “Advanced Quite Hot facilities”).
-replacefile2 ***"fn1" "fn2"***	New in 6.0. Define a replacement file. Files with name *fn1* (path is ignored) are replaced with *fn2*. (See manual under “Advanced Quite Hot facilities”).
-replacefileds ***"fn" "ds"***	New in 6.0. Define a replacement file. Files with name *fn* (path is ignored) are replaced with dataset *ds*. (See manual under “Advanced Quite Hot facilities”).
-dataset:*ds* ***"filename"***	New in 6.0. Define a dataset called *ds* as being the file *filename.* (See manual under “Advanced Quite Hot facilities”).
-User:*name* ***"value"***	New in 5.0. This sets user variables that can be used in Stick On Text & Numbers, and in Condition commands, to control processing. Any name or number can be used containing letters and numbers. The names 1, 2 and 3 are built in to the user interface and easier to use, but you are not limited to these names.
-v:name ***"value"***	New in 6.0. Same as -User:name. Both set variables that can be used with the new variable feature.

-vars ***"filename"***	New in 6.0. Read the file *filename* and set variables as found there. Can appear more than once and all variables are set. (See variables manual).
-exportvar ***"name"***	New in 6.0. Add *name* to the list of variables to export – this does not cause exporting to happen. A name of "*" causes all variables to be exported. (See variables manual)
-exportvartxt ***"filename"***	New in 6.0. Write a list of variables exported to *filename*. (See variables manual)
-exportmarker ***"value"***	New in 6.0. Write a list of variables exported to output, preceding each line with *value.* Note that, according to platform, this string will be followed immediately by the variable and value (no space), and the line may be terminated by a carriage return (\r), a new line (\n), or both. (See variables manual).
-jobname ***"value"***	New in 6.0. Set the jobname, which can be picked up from the JOBNAME() function in variables, and defaults to the source filename. (See variables manual).
-varpattern ***"value"***	Used to set variables from the source filename. This is the same form as a filter string, but has no effect on whether files are processed. If the pattern does not match, no variables are set. (See variables manual)
-mdremovebefore ***"value"***	Remove some Quite-related metadata before processing the file. Options for *value*, which can be combined into a single string: a – remove all available below (and possibly future options) f – remove file attachments (containing XML commands). Does not remove the commands, just visible file attachments. c – remove all stored commands (XML) in PDF. This means the only commands listed are the new commands from this session, rather than the old and the new together. p – remove page info. Page info stores filename and page number when pages are processed, so that reordering files will keep the old numbering. Removing the page info means the new name and numbering will be used
-mdremoveafter ***"value"***	Same options as -mdremoveafter. If you remove stored commands (a or c option) it will no longer be possible to

see which commands were used to make the result. If you only remove file attachments (f option) the commands will be visible to Quite Imposing Plus and Quite Hot Imposing, but not simply extracted as an attachment. If you remove page info (a or p option) the original filenames will not be visible.

Additionally, where the command line supports more than one language, you can use the **-lang** option e.g. **-lang fr** to choose French if it is available. This was introduced in version 1.1.

Combining files

A new feature of Quite Imposing 4.0 is its ability to combine files. You can give multiple source files. These files are simply combined into a single PDF, which is then used for all the commands, as if it were a single file. You can specify multiple files in two different ways, which can be mixed.

-source *filename* can be repeated. Each new filename is added to the end of the list of files to be processed (combined in the order specified).

-source *directory* can also be used, specifying a directory name in place of a file name. The PDF files in the directory are combined.

When processing a directory the following notes apply.

- Only files whose names end **.pdf** are processed.
- If any file cannot be included, for example because it is secure, the entire process fails.
- The order of files is alphabetical, but allowing for numbers. To get the order we split each name up on the dots and numbers in it. Where files have numbers in the same position they are compared. Otherwise the split pieces are compared alphabetically. Leading zeroes are ignored. We ignore the difference between upper and lower case in comparisons. An example of ordering:
  a.pdf a2.pdf A3A.pdf a03B.pdf a22.pdf a23x6.pdf a23x55.pdf a24.pdf a200.pdf
- This is similar but not identical to the default order shown by both Windows Explorer and macOS Finder[2]..
- Where multiple **-source** parameters are used, each directory is processed, then the files are added to the list at the end. (Rather than sorting all the file names together, from different directories).

[2] Windows and macOS use slightly different rules. It also changed between versions of Windows. The rules used by Quite Hot Imposing are the same whether on macOS or Windows. For maximum portability it is probably best to avoid mixing upper and lower case characters except normal unaccented letters. For instance avoid assuming “é” and “É” will be sorted the same.

- The hot folder monitor may process XML files that it finds in the directory, but the command line will only use the XML file specified on the command line.

Splitting files

A new feature in Quite Hot Imposing 5.0 is the "Split/merge (partials)" command. This can optionally write multiple files, so the meaning of the -target option has been changed.

- It is an option in the Split/merge (partials) command whether to write multiple files. If the option is not selected, -target has its normal meaning
- If the option is chosen to write multiple files, the user can select the form of output file names.
- The filename part of -target (if any) will be ignored when writing multiple files. It is used only to select a directory to write.
- The default for Split/merge (partial) is to create another directory in the target, and write all files there. The name of this directory, is taken from the input file. For example, if the input file were ATLANTIC.PDF and the target was C:\HOT\OUT then a directory C:\HOT\OUT\ATLANTIC would be created. The command sequence can turn off this option and write multiple files to the target directory itself.
- The XML file also specifies the form of output file names; these always include the input file name (less .PDF) and a part number.

Security of splitting files

With the Split/merge command we see the ability of an XML file to specify a path name to write. To protect the system the following rules are applied:

1. The path name to write is always relative to the directory in the **-target** command line parameter. It cannot be absolute (for example in Windows it cannot start C:\ or \\, and in Mac it cannot start /).
2. No directory name starting with a dot (.) will be written.
3. The form .. is therefore forbidden, so a path cannot refer to a parent directory.

Default locations

If installed to the default locations, on an English language system, you will find the command line in the following locations for 5.0 or as notes. (Note: demo versions are not normally shipped any longer).

- Windows, live

c:\Program Files\Quite\Quite Hot Imposing 5.0\qi_applycommands.exe

- Windows, demo 3.0 (no longer used)

c:\Program Files\Quite\Quite Hot Imposing 3.0 Demo\qi_applycommands_demo.exe

- macOS, live, versions from 5.0

/Applications/Quite Hot Imposing 5.0/Quite Hot Imposing.app/Contents/MacOS/Quite Hot Engine – note that the Quite Hot Engine.app bundle continues to exist and work for compatibility reasons, but macOS now expects to find all app executables in one bundle.

- macOS, live, versions up to 4.0

/Applications/Quite Hot Imposing 4.0/Quite Hot Engine.app/Contents/MacOS/Quite Hot Engine

- macOS, demo 3.0 (no longer used)

/Applications/Quite Hot Imposing 3.0 Demo/Quite Hot Engine (demo).app/Contents /MacOS/Quite Hot Engine (demo)

If you want to get this information programmatically (registry in Windows, Launch Services API in macOS) please contact us for details.

Error and warning messages

Error and warning messages will be written to standard error. UTF-8 encoding is used.

The use of UTF-8 (a form of Unicode) has potentially important implications, especially when using a version in a language other than English. It can largely be ignored with the English version, since UTF-8 is the same as the Windows or macOS character set for all the unaccented letters, numbers and common punctuation.

In Windows, UTF-8 files written to a TXT file can usually be recognised as such and opened by NotePad or Word.

Lines starting ** (two asterisks) can be considered as an error message. Lines starting -- (two dashes) can be considered to be notes, which do not mean there is a problem.

Scripting samples

Sample Windows BAT file

This is an example of a BAT file in Windows. This BAT file always applies the XML rules in c:\tmp\booklet maker.xml, and always writes the result file to an existing directory c:\tmp\results with the same name as the input. This (unsupported) script requires a single parameter, the file name to process.

```
set source=%1%
if not defined source echo Error: Missing source file name
@if not defined source @goto done

set targetdir="c:\tmp\results"
if not exist %targetdir% echo Error: %targetdir% does not exist
@if not exist %targetdir% @goto done

set ctlfile="c:\tmp\booklet maker.xml"

set livecmd="c:\Program Files\Quite\Quite Hot Imposing 5.0\qi_applycommands.exe"
set cmd=%livecmd%
```

```
%cmd% -control %ctlfile% -source %source% -target %targetdir%
:done
```

Sample macOS Shell Script

This is an example of a shell script for macOS. This always applies the XML rules in /tmp/booklet maker.xml, and always writes the result file to the existing directory /tmp/results with the same name as the input. This (unsupported) script requires a single parameter, the file name to process.

```
#!/bin/sh
source="$1"
if [ "$source" == "" ] ; then
        echo "Error: Missing source file name"
        exit 1
fi

targetdir="/tmp/results"
if [ ! -d "$targetdir" ] ; then
        echo "Error: $targetdir does not exist"
        exit 1
fi

ctlfile="/tmp/booklet maker.xml"

app="/Applications"
liveapp="$app/Quite Hot Imposing 5.0/Quite Hot Imposing.app"
livecmd="$liveapp/Contents/MacOS/Quite Hot Engine"
cmd="$livecmd"

"$cmd" -control "$ctlfile" -source "$source" -target "$target"
```

AppleScript

macOS users working in AppleScript can use the **do shell script** command to run command lines. There are no special considerations to this particular command, but note the requirement for quoting most file names on the command line.

It is possible to run a shell script (see above) by its full name, or to generate and run the command line directly.

This (unsupported) fragment shows setting the source file from the AppleScript item **infile**. You would need to also set target and control options.

```
set quote to "'"
set space to " "
set installfolder to "/Applications/Quite Hot Imposing 6.0"
set exefolder to installfolder & "/Quite Hot Engine.app/Contents/MacOS "
set executable to exefolder  & "Quite Hot Engine"
```

```
set sourcepath to the quoted form of the POSIX path of infile
set commandline to quote & executable & quote & space
set commandline to commandline & "-source" & space & sourcepath & space
set commandline to commandline & ... add other options here ...
set resultstring to do shell script commandline
```

Technical support

We are of course happy to help if this information is not clear or if the command line tool does not seem to be working as expected.

Unfortunately, we are not able to offer programming support, nor assist you with the writing of scripts or programs that make use of this command line. We recommend the command line for experienced programmers and systems integrators only. The hot folder tool is designed to be easy to set up and use without special programming skills.

Quite Hot Imposing SDK

A more detailed documentation set is available with information designed to assist more advanced integrators or developers intending to use Quite Hot Imposing from programs as part of a integrated workflow solution. Topics include checking licensing, locating the executable, and pre-vetting control files. Please contact Quite Software for details.