Miramo®

Automated Publishing

mmServer Guide

VERSION 9.2

Readers of this documentation should note that its contents are intended for guidance only, and do not constitute formal offers or undertakings.

'License Agreement'

This software, called Miramo, is licensed for use by the user subject to the terms of a License Agreement between the user and Datazone Ltd. Use of this software outside the terms of this license agreement is strictly prohibited. Unless agreed otherwise, this License Agreement grants a non-exclusive, non-transferable license to use the software programs and related documentation in this package (collectively referred to as Miramo) on licensed computers only. Any attempted sublicense, assignment, rental, sale or other transfer of the software or the rights or obligations of the License Agreement without prior written consent of Datazone shall be void. In the case of a Miramo Development License, it shall be used to develop applications only and no attempt shall be made to remove the associated watermark included in output documents by any automated method.

The documentation accompanying this software must not be copied or re-distributed to any third-party in either printed, photocopied, scanned or electronic form.

The software and documentation are copyrighted. Unless otherwise agreed in writing, copies of the software may be made only for backup and archival purposes. Unauthorized copying, reverse engineering, decompiling, disassembling, and creating derivative works based on the software are prohibited. This notice is provided for information only, and does not constitute a License Agreement.

Datazone does not warrant that the software will be free from error or will meet your specific requirements. You assume complete responsibility for decisions made or actions taken based on information obtained using the software. Any statements made concerning the utility of the software are not to be construed as unexpressed or implied warranties.

Trademarks

FrameMaker, FrameMaker+SGML, Acrobat, Adobe, PostScript, Adobe Illustrator and TIFF are trademarks of Adobe Systems Inc. Macintosh is a trademark of Apple Computer Inc. Windows NT is a trademark of Microsoft Corporation.

PANTONE is a registered trademark of Pantone Incorporated.

Miramo is a Registered Trademark of Datazone Ltd (see above).

Miramo includes software developed by the Apache Software Foundation (http://www.apache.org/).

Datazone Limited
Palm Gate, Greenane,
Killarney, Co. Kerry, Ireland.
Tel: +353 64 66 289 64
Fax: +353 64 66 289 65
Email: miramo@datazone.com
www.miramo.com

13.09.17 920p21

User & Reference Guides

Contents

User Guide

Reference Guide

mmDraw Drawing Guide

mmServer Guide

mmChart Inline Charting Guide

Character Reference Guide

mmServer Guide

Contents

CHAPTER 1
Introduction

Overview

This *mmServer Guide* describes the following components.

1. The mmServer service that acts as a miramo job manager and a FrameMaker and Acrobat Distiller session manager.
2. The rmmcmd and mmConnect secure sockets remote processing interface.
3. The mmVisor GUI which provides an interface for managing, configuring and interrogating an instance of the mmServer service, running locally or on a remote host.

This document describes the command line interfaces (CLI) to the mmServer and mmConnect services. The mmVisor GUI provides a superset of the of the functionality of the command line interface.

rmmcmd is used to request Miramo jobs to be processed on a remote host, and also has a command line interface which is documented here.

The equivalent functionality to mmServer, mmConnect and rmmcmd is available via the Miramo API using C++, .NET or Java.[1]

Examples of using the API are included throughout this document.

The API examples included in this chapter assume that a connection "conn" to an instance of mmServer has been established using the following code:

C++
```
IServerConnectionManager * connMgr = ::GetServerManager();
IServerConnection * conn = connMgr->createServerConnection();
```

C#
```
ServerConnectionManager scm = ServerConnectionManager.GetServerManager();
ServerConnection conn = scm.CreateServerConnection();
```

Java
```
ServerConnectionManager scm = ServerConnectionManager.getServerManager();
ServerConnection conn = scm.createServerConnection();
```

This connection may be used as-is to interact directly with a local instance of mmServer.

To establish a connection to an instance of mmServer running on a remote host, the host name, user name and password supplied as connection parameters. The "username" and "password" must specify a valid Windows user account on "remotehost". Below is an example of code to switch the connection to mmServer running on "remotehost":

C++
```
conn->setHostName(L"remotehost");
conn->setUserName(L"username");
conn->setPassword(L"password");
```

1. The mmVisor application is itself written using the Miramo API.

C#
```
conn.HostName = "remotehost";
conn.UserName = "username";
conn.Password = "password";
```

Java
```
conn.setHostName("remotehost");
conn.setUserName("username");
conn.setPassword("password");
```

To switch the connection back to the local instance of **mmServer**, set the hostName to "localhost", or the local host name:

C++
```
conn->setHostName(L"localhost");
```

C#
```
conn.HostName = "localhost";
```

Java
```
conn.setHostName("localhost");
```

The *username* and *password* values are still present as connection properties, but are ignored for local connections.

The **miramo** API functions are documented in HTML format:

- C++ API documentation: %MM_HOME%\api\C++\html\index.html.
- Java API documentation: %MM_HOME%\api\java\html\index.html.
- .NET API documentation: %MM_HOME%\api\dotnet\html\index.html.

Source code for the examples is included in <installdir>/api/samples/language/mmServer-Control.

Users should check section **C++, Java and .NET job processing APIs** under **SYSTEM REQUIREMENTS** on page RN-v in the *Miramo Release Notes* before developing an API application.

CHAPTER 2

mmServer control and status reporting

Overview

The mmServer service is a Windows NT service that functions as a FrameMaker session manager, handling requests from the miramo program and APIs, and a job control and information manager. mmServer acts in two primary, distinct modes:

1. As a service, managing and processing job requests from miramo command line and API functions. The mmServer service also maintains operating statistics and job status information and supports a set of command-line options and API functions for reporting such statistics and information. These options are described in the following sections: mmServer **status options**, pages MS-7–8 and mmServer **job status options**, MS-9–13, and are accessible to all users.
2. The mmServer program also supports a set of options for starting and stopping the mmServer service, as well for job control. These options are described in the following sections: mmServer **service starting and stopping options**, page MS-6 and mmServer **job termination**, page MS-13.

   These options require Administrator user privileges.

Most of the functionality described in this chapter may be achieved more conveniently using the mmVisor GUI interface, which has largely superceded the command line interface described here.

mmServer API functions

Equivalent capabilities are also available via the remote processing rmmcmd command line interface (see CHAPTER 4, **Using rmmcmd**, on pages R-23–30), and via the C++, .NET and Java APIs.

The remainder of this chapter describes mmServer command line options, but includes API code samples where appropriate.

mmServer command line options summary

A summary of the command-line syntax for the mmServer program is output by the command: mmServer -help. The '-help' output is shown in Example 2.1.

All the mmServer command line options shown in Example 2.1 may be in mixed upper or lower case. I.e. '-Help' is equivalent to '-help', '-start' is equivalent to '-StArT' and so on.

mmServer **help options**

The mmServer help options provide a usage summary of all the mmServer control, operating mode and reporting options. The output from the help options is written to standard error, or to the file specified as an optional argument to '-help' (see Example 2.2 on page MS-5).

-help [*filename.txt*]

Output a usage summary, as shown in Example 2.1.

Running:

```
mmServer -help
```

outputs:

```
mmServer:  Miramo resource manager (vs 9.2.0p26)

Usage: mmServer

Option names are case-insensitive.

   [-h [filename]]
      Write this help message to stderr
      E.g. mmServer -h

      To include the output in a file, add name of file
      E.g. mmServer -h "mmServerHelp.txt"

   [-v]
      Write version info to stderr

   [-diag 0¦1¦2]
      Write additional diagnostic information to log file (0 = none, 1 = verbose, 2 = extreme)

   [-showEnv [filename]]
      Write mmServer environment data to stdout

      To include the output in a file, add name of file
      E.g. mmServer -checkenv "mmCheckEnv.txt"

   [-status]
   [-statusn]
      Write mmServer status summary to stdout
      Use -status for space separated attributes
      Use -statusn to include newlines between attribute values
      Use output re-direction to include the output in a file

      E.g. mmServer -status  > "mmServerStatus.txt"


   [-wlist [N] "keyoptions"]
      Write list of 'N' waiting jobs to stdout
      Order: descending, by longest wait time.
      Include key options (use '-wlist help' for key option description):
         all pid sta jid mpt ifn ifp ifs ofn ofp ofs cno sdt wtm cmd pgc
      'N' defaults to all waiting jobs
      Default: all

      E.g.: mmServer -wlist "sdt jid"
      lists the starting time and user job id of all waiting jobs

   [-plist [N] "keyoptions"]
      Write list of 'N' processing jobs to stdout
      Order: descending, by longest processing time.
      Include key options (use '-plist help' for key option description):
         all pid sta jid mpt ifn ifp ifs ofn ofp ofs cno sdt wtm ptm cmd
```

```
    'N' defaults to all processing jobs
    Default: all

[-clist [N] "keyoptions"]
    Write list of 'N' completed jobs to stdout
    Order: ascending, by most recently completed.
    Include key options (use '-clist help' for key option description):
        all pid sta stc rtc jid mpt ifn ifp ifs ofn ofp ofs cno sdt wtm ptm edt cmd pgc
    'N' defaults to value specified by -clistnumrecords option (see below)
    Default: all

[-killJob N]
    Kills the job identified by job number 'N' (available via jobNum attribute)

The following options require Administrator privileges:

[-start]
    Start the mmServer service. The service must already be
    registered.

[-stop [wait ¦ now]]
    Stop mmServer service. Either now, or wait for all currently
    processing jobs to finish. Jobs waiting but not processing
    are always discarded.
    Default: wait

[-runAs "username"]
    User name for the runAs user

[-password password]        Password for the runAs user
```

Example 2.1 Output from mmServer -help

The output of the 'help' option may be directed to a text file using the format shown in Example 2.2.

```
mmServer -help  filename.txt
```

Example 2.2 Sending <mmServer> -help output to a text file

-clist help

Output help for the mmServer '-clist' option. See the description of the '-clist' option (Example 2.6 on page MS-10).

-plist help

Output help for the mmServer '-plist' option. See the description of the '-plist' option (Example 2.9 on page MS-12).

-wlist help

Output help for the mmServer '-wlist' option. See the description of the '-wlist' option (Example 2.10 on page MS-13).

mmServer service registration options

Service registration options require Administrator privileges. The mmServer service must be stopped before using service registration options. mmServer registration and RunAs user setup are performed automatically during product installation. The easiest way to change the RunAs user is to uninstall and re-install Miramo. Assuming correct initial installation there is no need to use the '-register' and '-unregister'options.

-register [auto ¦ manual]

Register the mmServer service in the Services Control Manager ServicesActive database.

If the auto parameter follows the '-register' option, the mmServer service will start every time the host system is booted. This is the default.

If the manual parameter follows the '-register' option, the mmServer service will start only when the '-start' option is used.

The '-runAs' and '-password' RunAs user options are always required when using the '-register' option.

-runAs *username*

User name for the RunAs user.

The '-runAs' option, and the '-password' option (see above), must always be used when the '-register' option is used. The '-runAs' and '-password' options are ignored if the '-register' option is not used.

The RunAs user, *username*, must have the 'Log on Locally' user right assigned.[1] This done automatically during installation.

-password *password*

Password for the RunAs user, specified using the '-runAs' option. If there is no password for the RunAs user then *password* is the null string, i.e. a pair of double quotation marks with nothing in between them, *viz*. "". '-pwd' is an alias for '-password'.

The '-password' option, and the '-runAs' option (see below), must always be used when the '-register' option is used. The '-password' and '-runAs' options are ignored if the '-register' option is not used.

-unregister

Unregister the mmServer service from the Service Control Manager ServicesActive database. The service will be unregistered only if it has previously been stopped using the '-stop' option, or the '-stop' option is used concurrently with the '-unregister' option.

mmServer service starting and stopping options

Starting and stopping options require Administrator privileges.

-start

Start the mmServer service. The service will start only when the '-start' option is used, or when the host system is re-booted if the '-register' option is set to auto.

-stop [wait ¦ now]

Stop the mmServer service. The service will stop only when the '-stop' option is used, or when the host system is shut down.

1. This is the SeInteractiveLogonRight, described in Microsoft article ID 279664 How to Set Logon User Rights with the Ntrights.exe Utility

The '-stop' option terminates the FrameMaker and Acrobat Distiller processes associated with the mmServer service, as well as the service itself.

If the wait parameter follows the '-stop' option, the mmServer service stops only after all the currently processing jobs have been completed. Use the now parameter to force an immediate shutdown of mmServer and all processes associated with it.

The equivalent code for using the Miramo API to stop and start the service is shown below:

C++
```
conn->control()->stopServer();
conn->control()->startServer();
```

C#
```
conn.Control.StopServer();
conn.Control.StartServer();
```

Java
```
conn.control().stopServer();
conn.control().startServer();
```

mmServer **status options**

Status options may be used by any system user. Status option output is written to standard output.

-status [help ¦ *keyoptions*]

Output the operational status of the mmServer service.

Running the command:

```
mmServer -status
```

produces output identical to that produced by the '-statusn' output (see Example 2.5 on page MS-9) except that the '-status' output has no line break and indent formatting.

The output of the '-status' option may be directed to a text file as illustrated in Example 2.3.

```
mmServer -status > filename
```

Example 2.3 Sending -status output to a file

If the '-status' option is followed by the word help help information is output to standard error, e.g.

Running:

```
mmServer -status help
```

outputs the following '-status' help information to standard error:

```
mmServer:  Miramo resource manager (vs 9.2.0p26)

Include keys for -status:


The following -status key parameters select output attributes.

help   Display this help message
all    All possible values
tim    current time
tzn    current timezone
run    mmServer running? (Y or N)
ver    MEM version
fmv    FrameMaker version
```

```
sob     Start on boot setting (Y or N)
ras     RunAs user
umo     User mode
prt     Default printer
osr     Operating system release
osn     Operating system name
osv     Operating system version
cpn     CPU name
cpc     Number of physical CPUs
hid     mmHostID
fpa     Full path of mmServer executable
log     Log to file (Y or N)
lff     Logfile folder
lfn     Logfile name
lmb     Maximum number of log file backups
lmr     Maximum number of job records to be logged before a log file is backed up
sct     Number of sessions
mpt     Setting for default max processing time.
muc     Max use setting
sdt     Start date and time of mmServer service
upt     Uptime, elapsed time since mmServer service last started
wjc     Waiting job count
rjc     Running job count
sjc     Succeeded job count since mmServer last started
fjc     Failed job count since mmServer last started
fwt     Total jobs failed due to wait timeout
fpt     Total jobs failed due to processing timeout
jno     Job number
pct     Record page count (Y or N)
tpc     Total number of PDF pages since mmServer last started.
```

Example 2.4 Output from mmServer -status help

-statusn

Output the operational status of the mmServer service. The '-statusn' option is designed for interactive use in a console window. Its output is as shown in Example 2.5, i.e. attributes are shown on separate lines.

Running the command:

```
mmServer -statusn
```

produces output in the in one or other of the following forms:

```
<mmServerStatus
    time="Sep 17 2013, 04:34:12.512"
    timezone="GMT Daylight Time"
    running="Y"
    status="2"
    totalChannels="5"
    maxChannels="10"
    currentTransitionState="7"
    totalTransitionStates="7"
    transitionStateDescription="Ready"
    mmServerExitCode="0"
    memVersion="9.2.0p26 (UPC+mmChart)"
    fmVersion="11.0.1p382"
    startOnBoot="N"
    runAs=".\mmUser"
    filePath="C:\ap\Miramo\bin\mmServer.exe /service"
    hostName="FOX"
    OS="Windows NT 6.1.7601"
    OSname="Windows Server 2008 R2 Enterprise"
    OSversion="Service Pack 1"
    CPU="Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz"
    HW="FUJITSU PRIMERGY TX120 S3p [Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz]"
    CPUcount="1 (8)"
    startTime="Sep 16 2013, 18:28:13.833"
```

```
    upTime="10 hours, 5 minutes, 58.679 seconds"
    waitingJobs="0"
    runningJobs="0"
    succeededJobs="171"
    failedJobs="5"
    failedWT="0"
    failedPT="0"
    totalPdfPageCount="86,410"
    jobNum="176"
    userMode="all"
    printer="Adobe PDF"
    tmpDir="C:\Users\mmUser\AppData\Local\Temp"
    log="Y"
    logFolder="C:\ap\Miramo\logs"
    imageTmpDir="C:\ap\Miramo\tmp\imageCache"
    logNumRecords="1"
    logFileBackups="20"
    logMaxRecords="50000"
    spoolerRetries="10"
    clistNumRecords="200"
    defaultJoboptions="mmStandard"
    runAsPassword="f+oAnOPL74f5CIjw41meot4sMNjh09US"
    rsPeriod="24:00:00">
    <channelGroups>
        <channelGroup
            name="default"
            channels="5"
            maxProcTime="3000"
            maxRetries="3"
            countPdfPages="Y"
            maxUsageCount="1000"
            printer="Adobe PDF"
            succeededJobs="171"
            failedJobs="5"
            waitingJobs="0"
            failedWT="0"
            failedPT="5"
            pdfPageCount="86410"/>
    </channelGroups>
</mmServerStatus>

or

mmServer:Not registered
```

Example 2.5 Output from mmServer -statusn

The equivalent XML status data may displayed via the API as follows:

C++
```
IServerStatus status = conn->status();
std::wcout << status->xml() << std::endl;
```

C#
```
ServerStatus status = conn.Status;
Console.Out.WriteLine(status.Xml);
```

Java
```
ServerStatus status = conn.getStatus();
System.out.println(status.getXml());
```

mmServer job status options

Job status options may be used by any system user. Job status option output is written to standard output unless the `help` keyword is used, in which case the output is written to standard error.

-clist [*n* ¦ help] [*keyoptions*]

List the most recently completed job records. Most recent job records are output first. By default all in-memory job records for completed jobs are listed.

If the '-clist' option is followed by the word **help** help information is output to standard error, e.g.

Running:

```
mmServer -clist help
```

outputs the following '-clist' help information to standard error:

```
mmServer:  Miramo resource manager (vs 9.2.0p26)

Include keys for -clist:

The following key parameters select output attributes and values.

help   Display this help message
all    All possible values
cmd    Include entire command line in <jobRecord> element.
jno    Job number (unique)
pid    Process ID for job
jid    User Job ID. Null unless miramox -jobID "value" option used.
mpt    Max processing time for this job
ifn    Input file name (basename)
ifp    Input file path
ifs    Input file size
ofn    Output file name (basename)
ofp    Output file path
cno    Channel number used for job (1 to N)
sdt    Start date and time of miramox job
wtm    Waiting time
ptm    Processing time
sta    Status of job:
         S=success, FWT=failed due to wait timeout, FPT=failed due to processing timeout,
                    F=failed to process.
stc    Status code for job: 0 for success
rtc    Number of job retries
ofs    Output file size
pgc    Number of pages in output document, if in PDF format and -countpdfpages is Y
edt    Job completed date and time
```

Example 2.6 Output from mmServer -clist help

If the '-clist' option is followed by an *n* (integer) value, then *n* completed-job job records are output.

A restricted set of attributes is output by including one or more *keyoptions*.

Running:

```
mmServer -clist 1
```

outputs:

```
<jobList status="Completed" count="1">
<jobRecord
        jobNum="176"
        channelGroupNum="1"
        channelGroupName="default"
        pid="13068"
        threadId="12580"
        status="S"
        statusCode="0"
        retries="0"
        jobID=""
        maxProcTime="PT3000.0S"
```

```
        user="man900"
        host="golo"
        composer=""
        iFileName="snake8"
        iFilePath="\\3.0.0.12\\tmp\snake8"
        iFileSize="218"
        oFileName="snake8.pdf"
        oFilePath="\\3.0.0.12\\tmp\snake8.pdf"
        oFileSize="9101"
        pdfPageCount="1"
        channelNum="4"
        startDate="2014-09-17T01:49:47.775"
        waitTime="P000DT00H00M00.0S"
        procTime="P000DT00H00M04.433S"
        endDate="2014-09-17T01:49:52.208"
        > ... entire Miramo command line included here ... </jobRecord>
</jobList>
```

Example 2.7 Output from mmServer -clist

The output shown in Example 2.7 has been formatted to show status attributes on separate lines to enhance readability. The actual output from the '-clist' option is all on one line, except in the case the '-clist' option is followed by **help**.

Running:

```
mmserver -clist 2 "jid ifn pgc edt"
```

outputs selected data relating to the two most recently completed jobs:

```
1  <jobList status="Completed" count="2">
2          <jobRecord jobID="" user="man900" host="golo" composer=""
                      iFileName="snake8" pdfPageCount="1"
                      endDate="2013-09-17T01:49:52.208"/>
3          <jobRecord jobID="" user="man900" host="golo" composer=""
                      iFileName="spider8" pdfPageCount="1"
                      endDate="2013-09-17T01:49:42.459"/>
4  </jobList>
```

Example 2.8 Output from mmServer -clist "jid ifn pgc edt" (2)

The API code samples below illustrate how to select and format different items of information from a job record (job number and status code, equivalent to running mmserver -clist "jno stc"), as well as outputting the job record XML (mmserver -clist)

C++

```
// Get Jobs
const JobList * jobs = conn->completedJobs();
for(size_t i = 0; i < jobs->size(); i++) {
    std::wcout << (*jobs)[i]->jobNumber() << L" " << (*jobs)[i]->jobStatusCode()
        << std::endl;
    std::wcout << (*jobs)[i]->xml() << std::endl;
}
```

C#

```
// Get Jobs
IList<Job> jobList = conn.CompletedJobs;
foreach (Job job in jobList)
    {
    Console.Out.WriteLine(String.Format("Job {0}: {1}", job.JobNumber,
        job.JobStatusCode));
    Console.Out.WriteLine(job.Xml);
    }
```

Java

```
// Get Jobs
```

```
java.util.List<Job> jobList = conn.getCompletedJobs();
for (Job job:jobList) {
    System.out.println(String.format("Job %d: %s", job.getJobNumber(),
        job.getJobStatusCode()));
    System.out.println(job.getXml());
}
```

-plist [*n* ¦ help] [*keyoptions*]

Write status details of all currently processing jobs to standard output.

If the '-plist' option is followed by the word **help** help information is output to standard error, e.g.

Running:

```
mmServer -plist help
```

outputs the following '-plist' help information to standard error:

```
mmServer:  Miramo resource manager (vs 9.2.0p26)

Include keys for -plist:

The following key parameters select output attributes and values.

help   Display this help message
all    All possible values
cmd    Include entire command line in <jobRecord> element.
jno    Job number (unique)
pid    Process ID for job
jid    User Job ID. Null unless miramox -jobID "value" option used.
mpt    Max processing time for this job
ifn    Input file name (basename)
ifp    Input file path
ifs    Input file size
ofn    Output file name (basename)
ofp    Output file path
cno    Channel number used for job (1 to N)
sdt    Start date and time of miramox job
wtm    Waiting time
ptm    Processing time
```

Example 2.9 Output from mmServer -plist help

C++
```
const JobList * jobs = conn->processingJobs();
for(size_t i = 0; i < jobs->size(); i++) std::wcout << (*jobs)[i]->xml() << std::endl;
```

C#
```
IList<Job> jobList = conn.ProcessingJobs;
foreach (Job job in jobList) Console.Out.WriteLine(job.Xml);
```

Java
```
java.util.List<Job> jobList = conn.getProcessingJobs();
for (Job job:jobList) { System.out.println(job.getXml()); }
```

-wlist [*n* ¦ help] [*keyoptions*]

Write status details of all waiting jobs to standard output.

If the '-wlist' option is followed by the word **help** help information is output to standard error, e.g.

Running:

```
mmServer -wlist help
```

outputs the following '-wlist' help information to standard error:

```
mmServer:  Miramo resource manager (vs 9.2.0p26)

Include keys for -wlist:

The following key parameters select output attributes and values.

help   Display this help message
all    All possible values
cmd    Include entire command line in <jobRecord> element.
jno    Job number (unique)
pid    Process ID for job
jid    User Job ID. Null unless miramox -jobID "value" option used.
mpt    Max processing time for this job
ifn    Input file name (basename)
ifp    Input file path
ifs    Input file size
ofn    Output file name (basename)
ofp    Output file path
cno    Channel number used for job (1 to N)
sdt    Start date and time of miramox job
wtm    Waiting time
```

Example 2.10 Output from mmServer -wlist help

C++
```
const JobList * jobs = conn->queuedJobs();
for(size_t i = 0; i < jobs->size(); i++) std::wcout << (*jobs)[i]->xml() << std::endl;
```

C#
```
IList<Job> jobList = conn.QueuedJobs;
foreach (Job job in jobList) Console.Out.WriteLine(job.Xml);
```

Java
```
java.util.List<Job> jobList = conn.getQueuedJobs();
for (Job job:jobList) { System.out.println(job.getXml()); }
```

mmServer **job termination**

The following job termination option may be used by any system user.

-killJob jobNum

Terminate a job currently on the waiting or processing lists.

jobNum is a unique job number, distinct from jobID (see the **miramo** '-jobID' option on page R-6 in the *Miramo Reference Guide*). **mmServer** assigns every job a jobNum that is unique within an **mmServer** session. jobNum has an integer value that is incremented by 1 for every new job.

mmServer -killJob 22 is equivalent to:

C++ conn->killJob(22);

C# conn.KillJob(22);

Java conn->killJob(22);

mmServer **logging**

Job data and status messages are stored in the file mmServerLog.xml. By default this file is in the folder: %MM_HOME%\logs. The location of the mmServerLog.xml data and message logging file may be changed via the **mmVisor** GUI.

The mmServerLog.xml file contains a single root element, <mmsLogFile>, which includes zero or more of the following elements:

```
<jobRecord ... >
<msg ... >
```

Date, time and elapsed time information contained in the log file is presented as specified in the W3C 'Extended Log File Format', except that values relate to local time, not GMT.

<mmsLogFile ... /> record

An example of the <mmsLogFile ... /> record is given in Example 2.11.

```
<mmsLogFile
    time="2014-09-16T18:28:13.833"
    timezone="GMT Daylight Time"
    status="0"
    totalChannels="5"
    maxChannels="10"
    currentTransitionState="0"
    totalTransitionStates="7"
    transitionStateDescription=""
    mmServerExitCode="0"
    memVersion="9.2.0p26 (UPC+mmChart)"
    fmVersion="11.0.1p382"
    startOnBoot="N"
    runAs=".\mmUser"
    filePath="C:\ap\Miramo\bin\mmServer.exe /service"
    hostName="FOX"
    OS="Windows NT 6.1.7601"
    OSname="Windows Server 2008 R2 Enterprise"
    OSversion="Service Pack 1"
    CPU="Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz"
    HW="FUJITSU PRIMERGY TX120 S3p [Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz]"
    CPUcount="1 (8)"
    userMode="all"
    printer="Adobe PDF"
    tmpDir="C:\Users\mmUser\AppData\Local\Temp"
    log="Y"
    logFolder="C:\ap\Miramo\logs"
    imageTmpDir="C:\ap\Miramo\tmp\imageCache"
    logNumRecords="1"
    logFileBackups="20"
    logMaxRecords="50000"
    spoolerRetries="10"
    clistNumRecords="200"
    defaultJoboptions="mmStandard"
    runAsPassword="f+oAnOPL74f5CIjw41meot4sMNjh09US"
    rsPeriod="24:00:00
    />
```

Example 2.11 Format of <mmsLogFile ... /> record in mmServerLog.xml file

The <mmsLogFile ... /> record occurs once, at the beginning of the mmServerLog.xml file, and contains metadata pertaining to the file.

<msg ... > record

The format of the <msg ... > record is illustrated in Example 2.12.

```
<mmsLogFile
    time="2014-09-16T18:28:13.833"
    timezone="GMT Daylight Time"
    status="0"
    totalChannels="5"
    maxChannels="10"
    currentTransitionState="0"
    totalTransitionStates="7"
    transitionStateDescription=""
```

```
mmServerExitCode="0"
memVersion="9.2.0p26 (UPC+mmChart)"
fmVersion="11.0.1p382"
startOnBoot="N"
runAs=".\mmUser"
filePath="C:\ap\Miramo\bin\mmServer.exe /service"
hostName="FOX"
OS="Windows NT 6.1.7601"
OSname="Windows Server 2008 R2 Enterprise"
OSversion="Service Pack 1"
CPU="Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz"
HW="FUJITSU PRIMERGY TX120 S3p [Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz]"
CPUcount="1 (8)"
userMode="all"
printer="Adobe PDF"
tmpDir="C:\Users\mmUser\AppData\Local\Temp"
log="Y"
logFolder="C:\ap\Miramo\logs"
imageTmpDir="C:\ap\Miramo\tmp\imageCache"
logNumRecords="1"
logFileBackups="20"
logMaxRecords="50000"
spoolerRetries="10"
clistNumRecords="200"
defaultJoboptions="mmStandard"
runAsPassword="f+oAnOPL74f5CIjw41meot4sMNjh09US"
rsPeriod="24:00:00"
>... message contents ...</msg>
```

Example 2.12 Format of <msg ... > record in mmServerLog.xml file

<msg ... > records contain information about system management events. Any number of <msg ... > records may occur throughout the mmServerLog.xml file.

<jobRecord ... > record

The format of the <jobRecord ... > record is illustrated in Example 2.13.

```
<jobRecord
    jobNum="1"
    channelGroupNum="1"
    channelGroupName="default"
    pid="10824"
    threadId="10536"
    status="S"
    statusCode="0"
    retries="0"
    jobID=""
    maxProcTime="PT3000.0S"
    user="man100"
    host="fox"
    composer=""
    iFileName="aaa.mmx"
    iFilePath="C:\u\miramo\man\man100\mmtmp/aaa.mmx"
    iFileSize="892"
    oFileName="aaa.pdf"
    oFilePath="C:\u\miramo\man\man100\mmtmp/aaa.pdf"
    oFileSize="14436"
    pdfPageCount="1"
    channelNum="1"
    startDate="2014-09-16T18:32:24.969"
    waitTime="P000DT00H00M00.0S"
    procTime="P000DT00H00M01.359S"
    endDate="2014-09-16T18:32:26.328"
    > ... entire Miramo command line included here ... </jobRecord>
```

Example 2.13 Format of <jobRecord ... > record in mmServerLog.xml file

One <jobRecord ... > record is included in the mmServerLog.xml file for every Miramo job, up to a maximum specified by the -logMaxRecords option.

mmServer diagnostic options

Diagnostic option output is written to standard error.

-v

Output license information and **mmServer** version.

Running the command:

```
mmServer -v
```

produces output in the form:

```
mmServer:          Miramo resource manager (vs 9.2.0p26)

Expiry date:       (None)
Licensed hostid:   8e80306
Customer:          (Any)
License type:      Production
Serial number:     520
License version:   920
Max channels:      10
mmServer version:  9.2.0p26
Licensed modules:  miramo mmChart
```

-showEnv *filename*

Use the '-showEnv' option to specify the name of a file for **mmServer** output logging details of its operating environment and system configuration.

filename may be the full, absolute path name of a file, i.e a file name beginning with a drive letter, or '\\' to denote UNC nomenclature, provided the folder is writable by the **mmServer** runAs user. Otherwise *filename* is interpreted relative to the current folder.

The '-showEnv' option may be used with either on its own or with combination of command line options. If **mmServer** is not running the '-showEnv' option will fail with an error message.

Starting the mmServer service

The **mmServer** service must be started before Miramo can be used. It must be started by a user having Administrator privileges.

If the **mmServer** service has been stopped using the '-stop' option and has not been unregistered, it can be started as shown in Example 2.14.

```
mmServer -start
```

Example 2.14 Starting mmServer

Stopping the mmServer service

The **mmServer** service may be stopped by a user having Administrator privileges or by the RunAs user, by typing the command shown in Example 2.15 in a console window.

```
mmServer -stop
```

Example 2.15 Stopping mmServer

The command shown in Example 2.15 stops the mmServer service and terminates the FrameMaker and Acrobat Distiller sessions associated with mmServer, as well as the mmServer service itself. Unless the '-stop' option is followed by the now parameter mmServer will wait for currently executing jobs to complete before terminating.

The mmServer must be stopped before it can be unregistered.

After the mmServer service has been stopped the miramo command will no longer work.

Unregistering the mmServer service

The mmServer service may be removed from the Service Control Manager ServicesActive database by using the '-unregister' option. This is illustrated in Example 2.16.

```
mmServer -unregister
```

Example 2.16 Unregistering mmServer (1)

The '-unregister' option may only be used by a user having Administrator privileges.

The '-unregister' option is effective only if the mmServer service is currently stopped.

The command shown in Example 2.17 both stops and unregisters the mmServer service.

```
mmServer -stop -unregister
```

Example 2.17 Unregistering mmServer (2)

The command shown in Example 2.17 is equivalent to the commands shown in Example 2.18.

```
mmServer -stop
mmServer -unregister
```

Example 2.18 Unregistering mmServer (3)

Unregistering mmServer may be done by calling the generic mmServer 'executeMmServerRun' method as follows:

C++
```
std::wstring dataOut;
conn->executeMmServerRun(L"-unregister", dataOut);
```

C#
```
String dataOut="";
conn.ExecuteMMServerRun("-unregister", ref dataOut);
```

Java
```
executeMMServerRun("-unregister");
java.lang.String dataOut = getLastOutput();
```

After calling the executeMmServerRun method, dataOut will contain any error message text generated by the call.

RunAs user account logon access

The following command may be used by a user with Administrator privileges to enable/disable logging on as the mmServer RunAs user.

-noLogon [Y ¦ N]

If '-noLogon' has a value of Y the RunAs user account will not appear on the welcome screen. This is the recommended setting.

mmServer **configuration file**

When the **mmServer** service starts up it reads a set of operating parameters from the file: %MM_HOME%\config\mmserver.xml. The format of the mmserver.xml configuration file is illustrated in Example 2.19.

```
<?xml version="1.0" encoding="utf-8"?>
<mmConfig
      userMode="all"
      printer="Adobe PDF"
      tmpDir="C:\Users\mmUser\AppData\Local\Temp"
      log="Y"
      logFolder="C:\ap\Miramo\logs"
      imageTmpDir="C:\ap\Miramo\tmp\imageCache"
      logNumRecords="1"
      logFileBackups="20"
      logMaxRecords="50000"
      spoolerRetries="10"
      clistNumRecords="200"
      defaultJoboptions="mmStandard"
      runAs=".\mmUser"
      runAsPassword="f+oAnOPL74f5CIjw41meot4sMNjh09US"
      startOnBoot="N"
      rsPeriod="24:00:00">
      <channelGroups>
            <channelGroup
                  name="default"
                  channels="5"
                  maxProcTime="3000"
                  maxRetries="3"
                  countPdfPages="Y"
                  maxUsageCount="1000"
                  printer="Adobe PDF"/>
      </channelGroups>
</mmConfig>
```

Example 2.19 mmServer configuration file

The **mmVisor** GUI application is used to change **mmServer** operating parameters. Whenever **mmServer** operating parameters are changed via the **mmVisor** Control tab **mmVisor** saves the new values to the mmserver.xml file. The new settings become effective only after **mmServer** is restarted.

CHAPTER 3

Configuring the mmConnect service

The mmConnect and rmmcmd sockets interface

The mmConnect and rmmcmd programs comprise server and client sides of a sockets interface that enables miramo commands to be run locally or remotely by the rmmcmd program, provided the mmServer service is running on the Miramo host machine.

The mmConnect program, which runs as a Windows NT service, listens on a TCP port for connections that originate from instances of the rmmcmd program running on remote hosts. When mmConnect receives a request from rmmcmd it authenticates the user name and password and runs the command initiated by rmmcmd from the remote host. All communications between rmmcmd and mmConnect are encrypted.[1]

This chapter gives information on mmConnect command line options for starting and stopping the mmConnect service, and for changing mmConnect configuration parameters (for example, the TCP/IP port number upon which mmConnect listens for connections from the client rmmcmd program).

The mmConnect and mmServer services must be installed and running on the Miramo host machine; both services are started automatically as part of the installation process. The rmmcmd program may be installed on any host that is running an operating system of any type supported by rmmcmd. For the rmmcmd program to work, it must be run on a networked host that can make a TCP/IP connection to the Miramo host machine.

mmConnect operates on hosts running Windows NT 5.0 or higher. rmmcmd operates on Windows NT hosts and on Sun Solaris, but may also be available for additional operating systems.[2]

rmmcmd is an executable command-line wrapper for rmmcmd.dll, described in CHAPTER 10, **rmmcmd.dll**, on pages MS-49–51.

---

1. The protocol used for sending messages between the client (rmmcmd) and server (mmConnect) uses standard cryptographic techniques and algorithms to ensure that data is protected from unwanted viewing and potential corruption or changing of data between source and destination.

In order to keep the data confidential, all data is encrypted using 3DES (Triple DES - Data Encryption Standard). This is known as 3DES as it performs 3 individual DES operations on the data, allowing the otherwise insecure DES operation to be used securely.

As data can be corrupted or changed, the data protocol includes a MAC, or Message Authentication Cryptogram, which is effectively a checksum of the data, encrypted using 3DES. This ensures that even if data is changed, the MAC cannot be changed, without knowing the encryption key.

Individual messages use a randomly generated key known as a Session Key. The session key is sent in the message encrypted under a Master Key. This Master Key is embedded in the software. By using a Session Key, the amount of data transmitted under a single key is reduced. The collection of large amounts of data encrypted under a single key is essential for virtually all kinds of Cryptographic attacks.

2. Contact support@datazone.com for an up to date list of the operating systems supported by rmmcmd.

mmConnect does not have an API.

mmConnect and rmmcmd are inoperative unless the mmConnect service is started during installation or activated by an Administrator user via console commands. See **Starting the mmConnect service** on page MS-22.

mmConnect -help option

The mmConnect service '-help' option provides a usage summary of all the mmConnect options. The output from the '-help' option is written to standard error.

-h[elp]

Running:

```
mmConnect -help
```

outputs:

```
mmConnect vs 9.2.0p26
: Network server application supplying remote access to mmServer.

Usage: mmconnect [options]

options:
  -help
      displays this help message
  -v
      display version information

  The following options require Administrator privileges:

  -cwd [dir]
      set the default working directory for the service
  -logfile filename
      specify filename for logging diagnostic information
  -maxthreads n
      specify maximum number of threads in thread pool
  -port [portnumber]
      portnumber is an integer representing the TCP listening port number
      for mmConnect
  -start ¦ -stop [ now ¦ wait ]
      start/stop the mmConnect server.
  -status ¦ -statusn
      write mmConnect status summary to stdout
      Use -status for space separated attributes
      Use -statusn to include newlines between attribute values
      Use output re-direction to include the output in a file
```

Example 3.1 Output from mmConnect -help

mmConnect service starting and stopping options.

Starting and stopping options require Administrator privileges.

-start

Start the mmConnect service.

-stop

Stop the mmConnect service.

-startOnBoot [Y ¦ N]

Start the mmConnect service automatically on system boot.

Default: Y

-v

Display mmConnect version on standard error.

Running the command:

```
mmConnect -v
```

produces output in the form:

```
mmConnect vs 9.2.0p26
```

mmConnect service operating mode options

Operating mode options require Administrator privileges. Operating mode options take effect when the mmConnect service is next started.

-port *portnumber*

The mmConnect '-port' option sets the primary TCP listening port number for mmConnect, to receive connections from rmmcmd (see the rmmcmd '-port' option on page R-25).

Note mmConnect actually listens on two ports. The second port number, used for status information, is allocated as: *portnumber* + 1. If either *portnumber* or *portnumber* + 1 is unavailable mmConnect will fail to start. In that case use the '-port' option with a different *portnumber* value.

The '-port' option may only be used with the '-start' option *or* with the '-status' option.

Default: 22084 (& 22085)

-cwd *dirname*

The mmConnect '-cwd' option sets the default working directory for file processing. This directory should have read, write and access permissions for all rmmcmd users (see page R-26).

If this directory does not have read, write and access permissions for a rmmcmd user, then the rmmcmd program must be run using the '-cwd' option followed by a path name of a directory which the rmmcmd login user does have read, write and access permissions.

Default: %TEMP%

-maxthreads *n*

The mmConnect '-maxthreads' option sets the maximum number of threads in the thread pool.

Default: 5

mmConnect status option

Status options may be used by any system user. Status option output is written to standard output.

-status

The mmConnect '-status' option displays the status of the mmConnect service.

Running the command shown in Example 3.2.

```
mmConnect -status
```

Example 3.2 Using mmConnect '-status' option

produces output in the form shown in Example 3.3.

```
<mmConnectStatus
        time="2014-09-17T04:34:20.150"
        running="Y"
        memVersion="9.2.0p14"
        startOnBoot="Y"
        runAs="LocalSystem"
        filePath="C:\ap\Miramo\bin\mmConnect.exe"
        workingDir="C:/ap/Miramo/tmp"
        port="22084"
        threads="5"
        startTime="2014-09-16T17:19:36.110"
        upTime="P000DT11H14M44.40S"
        />
```

Example 3.3 Output from: mmConnect -status

Starting the mmConnect service

Example 3.4 shows how to start the mmConnect service.

```
mmConnect -start
```

Example 3.4 Starting the mmConnect service (1)

The mmConnect service and the mmServer service must be started locally on the Miramo host prior to running rmmcmd. The command shown in Example 3.4 must be run by a user with Administrator privileges.

CHAPTER 4

Using rmmcmd

Overview

The rmmcmd client interacts with mmServer on a remote host via the mmConnect secure sockets gatekeeper service. The mmConnect service runs on the Miramo host machine, i.e. the same host as mmServer.

rmmcmd supports two types of operations, either locally or when the mmServer service is running on a remote host.

1. Starting, stopping and configuring the operating parameters of the mmServer service.

   These operations require the name and password of an Administrator user on the mmServer host, and the mmConnect service must be running on the mmServer host.

2. Executing Miramo jobs on a remote host.

   Executing jobs on a remote host requires the name and password of a user on the mmServer host, and the mmConnect and mmServer services must be running on the mmServer host.

mmConnect operates on hosts running Windows NT 5.0 or higher. rmmcmd operates on Windows NT hosts and on Sun Solaris, but may also be available for additional operating systems.[1]

rmmcmd is an executable command-line wrapper for rmmcmd.dll. Equivalent capabilities are available in the C++, .NET and Java APIs.

mmConnect and rmmcmd are inoperative unless the mmConnect service is running on the Miramo host machne. mmConnect is started automatically during product installation, or activated by an Administrator user via a console command (see **Starting the mmConnect service** on page MS-22) or via the mmVisor GUI (see **Starting the mmServer service** on page MS-32). .

The Miramo API provides equivalent functionality for remote queuing of miramo jobs, and remote administration of mmServer. Source code examples are included where appropriate.

rmmcmd help options

The rmmcmd help options provide usage summaries of all the rmmcmd options. Help option output is written to standard error.

1. Contact support@datazone.com for an up to date list of the operating systems supported by rmmcmd

-h[elp]

Running the command:

The output has the form shown in Example 4.1.

```
rmmcmd: Network client application for remote access to miramo.
Version 9.2.0b09

Usage: rmmcmd options [ miramo_options ¦ -mmServer mmserver_options ]

miramo_options:
  Options to be passed to miramo.
  use -mmhelp option to view more details

Note: The '-password' option must be used with
      all other options listed below, except for -help.

options:
  -cwd [dir]
      Override the default mmServer working directory
  -encrypt [0¦1]
      Set encryption to minimum (0) or maximum (1)
      Default: 0
  -help
      Display this help message
  -host hostname ¦ IP Address
      Hostname or IP address of the mmServer host
      Default: localhost ()
  -port portnumber
      Port number used by mmConnect on the mmServer host
      Default: 22084
  -mmhelp
      Displays miramo help
  -user
      User name on the mmServer host
      Default: current user ()
  -password
      Password for the user on the mmServer host
  -status [ mmConnect ¦ mmServer ]
      Display status for mmConnect (default) or mmServer
  -mmServer ...
      run mmServer on remote host, passing through all subsequent options
      for example, to display uptime and total pdf page count for mmserver on host 3.0.0.12:
      rmmcmd -host 3.0.0.12 -user mmUser -password mmuser -mmServer -status "upt tpc"
```

Example 4.1 Output from: rmmcmd -help

-mmhelp

Use the **rmmcmd** '-mmhelp' option to display **miramo -help** output on standard error.

```
rmmcmd -mmhelp -user  userName -host mmHostName -password userPassword
```

Example 4.2 Using the rmmcmd '-mmhelp' option

The output from Example 4.2 has the form shown in Example 4.3.

```
Usage: miramo -h ¦ -v ¦ [-batch] \
          [-maxproctime secs] [-jobID str] [-maxwaittime secs] \
                  [-cwd name] [-userapi name] [-sendEnv Y¦N]\
          [-tENC utf-8 ¦ wcp-1252 ¦ us-ascii ¦ iso-8859-1] [-stripInput tab¦nl¦all¦N] -priority
                normal¦below¦idle \
          [-Bopts ¦ -Oopts] [-PDFopts] [-Mopts] [-Popts] [-Topts] [-Xopts]\
                      [infile]

   Command line flags and options are:
   -h    display short help
```

```
-v    display version and licensing information
-batch inputfile
      batch process inputfile containing <Doc ..> codes
-cwd dir
      set current working directory
-jobID "idstr"
      Assign jobid "idstr" for logging job status
-processGroup <int> ¦ name
      Assign job to processing channel group, specified by name or
      number (integer > 0)
-maxproctime nsecs
      Timeout after nsecs processing time
-maxwaittime nsecs
      Timeout after nsecs waiting time

-M    use mmpp macro pre-processor
-C    use mmchart charting pre-processor
-userapi apiname
      run Frame API client 'apiname' on document
-dENC FR ¦ mb ¦ EUC-CN ¦ BIG5 ¦ SJIS ¦ EUC-KR ¦ UTF-8
      specify default text processing mode + default fonts
      (default = FR (frameMaker 7) or UTF-8 (frameMaker 8)
-stripInput tab ¦ nl ¦ all ¦ N
      strip binary tabs, newlines, or both from input (N=don't strip)
-priority normal ¦ below ¦ idle
      set process priority

For more help on -M (macro processor) -T (template)
-B (Book) -O (Output) -PDF (Acrobat), -P (Printing) -X (Xslt) options, use
   -Mhelp, -Thelp, -Bhelp, -Ohelp, -PDFhelp, -Phelp or -Xhelp options.
```

Example 4.3 Output from: rmmcmd -mmhelp

rmmcmd remote host options

In the case when rmmcmd is being run on a different host to the host running mmServer, the following options must always precede all other options.

-host *IP address* **|** *hostname*

IP address or node name of the remote Miramo host.

-port *portnumber*

The rmmcmd '-port' option sets the TCP port number. It must be same as that used by mmConnect on the remote Miramo host to receive connections from rmmcmd (see the mmConnect '-port' option on page MS-21).

Default: 22084

-user *username*

username is the login user name on the remote Miramo host.

-password *password*

password is the password for login user on the remote, Miramo host.

The options above are also available via the Miramo API:

```
C++   conn->setHostName(L"remotehost");
      conn->setUserName(L"username");
      conn->setPassword(L"password");

C#    conn.HostName = "remotehost";
```

```
        conn.UserName = "username";
        conn.Password = "password";

Java    conn.setHostName("remotehost");
        conn.setUserName("username");
        conn.setPassword("password");
```

rmmcmd job processing options

Miramo job processing options may be used following rmmcmd remote host options to run Miramo jobs on a remote host. When running jobs on a remote host the '-user' and '-password' options refer a user on the remote host.

-cwd *dirname*

Set the working directory to be used by mmConnect on the Miramo host for this job. Overrides the default set by the mmConnect '-cwd' option (see page MS-21).

The '-cwd' must precede all *miramo_options*.

miramo_options

Miramo options to be passed to the Miramo host. .

rmmcmd options for interacting with mmServer

Using the rmmcmd '-mmServer' option enables all mmServer operations to be performed using rmmcmd on a remote host, see CHAPTER 2, mmServer control and status reporting, on pages MS-3–18.

When running mmServer commands on a remote host the rmmcmd remote host options, '-user' and '-password', sometimes must refer to a user with Administrator privileges on the remote host, as described in CHAPTER 2, mmServer control and status reporting.

-mmServer *mmServer options*

Run mmServer commands remotely. The rmmcmd '-mmServer' option must follow the rmmcmd remote host options and be followed only by mmServer options.

Running the command:

```
rmmcmd -user mmUser -host mmHostName -password password -mmServer -help
```

Example 4.4 Using rmmcmd '-mmServer -help' option

Produces the same output as running the command:

```
mmServer -help
```

Example 4.5 Using rmmcmd '-mmServer -help' option

locally on the mmServer host (see Example 2.1 on pages MS-4–5).

Similarly, running the commands:

```
rmmcmd -user mmUser -host mmHostName -pwd pwd -mmServer -status
rmmcmd -user mmUser -host mmHostName -pwd pwd -mmServer -clist
```

Example 4.6 Using rmmcmd '-mmServer -clist' and '-mmServer -status' options

produces output equivalent to running the commands:

```
mmServer -status
mmServer -clist
```

locally on the mmServer host (see Examples 2.5 and 2.6 on pages MS-9–10).

rmmcmd diagnostic option

Diagnostic option output is written to standard error.

-v

Running the command:

```
rmmcmd -v
```

produces output in the form:

```
rmmcmd: Network client application for remote access to miramo.
Version 9.2.0pn
```

-encrypt 0 | 1

Use the rmmcmd '-encrypt' option to set the encryption level.

A value of 0 applies encryption only to the user name, password and miramo command line. A value of 1 applies encryption to all data, including remote command options.

Default: 0

rmmCmd status options

Status options may be used by any system user. Status option output is written to standard output.

-status mmconnect

The rmmcmd 'status' option reports the operating status of the mmConnect service on the '-host' *hostname* mmServer host to standard error.

If the rmmcmd '-status' option is used with the mmConnect argument as shown in Example 4.7.

```
rmmcmd -status mmConnect -user mmUser -host mmHostName -password password
```

Example 4.7 Using rmmcmd '-status mmConnect' option

The output has the form shown in Example 4.8.

```
<mmConnectStatus
        time="2014-09-17T04:34:29.739"
        running="Y"
        memVersion="9.2.0p14"
        startOnBoot="Y"
        runAs="LocalSystem"
        filePath="C:\ap\Miramo\bin\mmConnect.exe"
        workingDir="C:/ap/Miramo/tmp"
        port="22084"
        threads="5"
        startTime="2014-09-16T17:19:36.110"
        upTime="P000DT11H14M53.629S"
        />
```

Example 4.8 Output from: rmmcmd -status mmConnect

The output shown in Example 4.8 is equivalent to the output shown in Example 3.3 on page MS-22. The difference is that the output produced in Example 3.3 is

from running mmConnect '-status' on the MEM host, whereas the output from Example 4.8 is from running mmConnect '-status' on a a remote host.

-status mmServer

The rmmcmd 'status' option reports the operating status of the mmServer service on the '-host' *hostname* mmServer host to standard error.

If the rmmcmd '-status mmServer' option is used as shown in Example 4.9.

```
rmmcmd -status mmServer -user mmUser -host mmHostName -password password
```

Example 4.9 Using the rmmcmd '-status mmServer' option

The output has the form illustrated in Example 4.10.

```
<mmServerStatus
        time="2014-09-17T04:34:30.10"
        timezone="GMT Daylight Time"
        running="Y"
        status="2"
        totalChannels="5"
        maxChannels="10"
        currentTransitionState="7"
        totalTransitionStates="7"
        transitionStateDescription="Ready"
        mmServerExitCode="0"
        memVersion="9.2.0p26 (UPC+mmChart)"
        fmVersion="11.0.1p382"
        startOnBoot="N"
        runAs=".\mmUser"
        filePath="C:\ap\Miramo\bin\mmServer.exe /service"
        hostName="FOX"
        OS="Windows NT 6.1.7601"
        OSname="Windows Server 2008 R2 Enterprise"
        OSversion="Service Pack 1"
        CPU="Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz"
        HW="FUJITSU PRIMERGY TX120 S3p [Intel(R) Xeon(R) CPU E3-1230 V2 @
            3.30GHz]"
        CPUcount="1 (8)"
        startTime="2014-09-16T18:28:13.833"
        upTime="P000DT10H06M16.177S"
        waitingJobs="0"
        runningJobs="0"
        succeededJobs="172"
        failedJobs="5"
        failedWT="0"
        failedPT="0"
        totalPdfPageCount="86410"
        jobNum="177"
        userMode="all"
        printer="Adobe PDF"
        tmpDir="C:\Users\mmUser\AppData\Local\Temp"
        log="Y"
        logFolder="C:\ap\Miramo\logs"
        imageTmpDir="C:\ap\Miramo\tmp\imageCache"
        logNumRecords="1"
        logFileBackups="20"
        logMaxRecords="50000"
        spoolerRetries="10"
        clistNumRecords="200"
        defaultJoboptions="mmStandard"
        runAsPassword="f+oAnOPL74f5CIjw41meot4sMNjh09US"
        rsPeriod="24:00:00
        />
```

Example 4.10 Output from: 'rmmcmd -status mmServer'

The same output may be produced via the Miramo API as follows:

C++
```
conn->setHostName(L"mmHostName");
conn->setUserName(L"mmUser");
conn->setPassword(L"password");
const memapi::IServerStatus * status = conn->status();
std::wcerr << status.xml();
```

C#
```
conn.HostName = "mmHostName";
conn.UserName = "mmUser";
conn.Password = "password";
ServerStatus status = conn.Status;
Console.Out.WriteLine(status.Xml);
```

Java
```
conn.setHostName("mmHostName");
conn.setUserName("mmUser");
conn.setPassword("password");
ServerStatus status = conn.getStatus();
System.out.println(status.getXml());
```

Running rmmcmd on a remote host

The purpose of the mmConnect service is to enable Miramo jobs to be run from any host that has the rmmcmd program and has a TCP/IP network interface. The mmConnect and mmServer services must both be started on the Miramo host before rmmcmd can be run on a remote host. The relationship between rmmcmd and the Miramo host is illustrated in Figure 4.1.

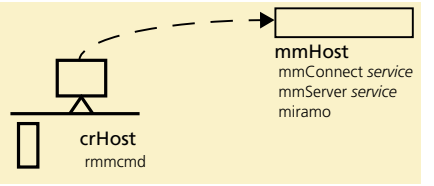

Figure 4.1 'crHost' controller and 'mmHostName' server

Unless data is transmitted or received via the rmmcmd standard input or standard output, all files referenced by rmmcmd have paths relative to the Miramo host, mmHostName.

rmmcmd examples (shared file systems)

Example 4.11 shows how to use the rmmcmd program to process a job on a remote Miramo host.

```
rmmcmd -host hostname -user username  -password password -Opdf T:\output.pdf T:\input.mmo
```

Example 4.11 Running rmmcmd (1)

In Example 4.11 *hostname* is the name of the Miramo host. The input file, 'T:\input.mmo', and the output file, 'T:\output.pdf', have file name paths relative to the remote Miramo host. UNC path naming may also be used.

All files referenced within the rmmcmd input, that is markup that contains 'file' options, e.g., the <Image ... /> code 'file' option, have paths relative to the Miramo host, *hostname*.

The equivalent Miramo API code is as follows:

C++
```
conn->setHostName(L"hostname");
conn->setUserName(L"username");
```

```
conn->setPassword(L"password");
IJobProcessor * pJobProcessor = conn->jobProcessor();
pJobProcessor->CreatePdf(L"T:\\input.mmo", L"T:\\output.pdf");
```

C#

```
conn.HostName = "hostname";
conn.UserName = "username";
conn.Password = "password";
JobProcessor processor = conn.JobProcessor;
processor.CreatePdf(@"T:\input.mmo", @"T:\output.pdf");
```

Java

```
conn.setHostName("hostname");
conn.setUserName("username");
conn.setPassword("password");
JobProcessor processor = conn.getJobProcessor();
processor.createPdf("T:\\input.mmo", "T:\\output.pdf");
```

Note that the @ in the C# example above prevents having to double-backslash the '\' character in filenames, as is necessary in the C++ and Java examples.

CHAPTER 5

mmVisor graphical interface

mmVisor overview

The mmVisor graphical interface displays the status and settings information described below.

- mmServer status and settings, for example:
  - Start time
  - Uptime
  - Total succeeded/waiting/failed job counts
  - RunAs user
  - Total number of processing channels
- mmServer operating environment, for example
  - Miramo / FrameMaker versions
  - Operating system
- mmServer processing channel groups status and settings, for example
  - Channel group name
  - Default printer
  - Succeeded/waiting/failed job counts
- mmServer job status information. For example, all of or any subset of the following:
  - Job number and user-defined job ID.
  - Calling options
  - Input file path and name
  - Output file path and name
  - Number of PDF pages in the output document
  - Job start time, waiting time and processing time
  - Job status
  - Maximum processing allowed for the job

  See pages MS-9–13.

In addition mmVisor enables users to:

- Terminate waiting and currently processing jobs
- View job input and output files using a single mouse click.
- View jobRecord data in logfiles in a human-readable format.

mmVisor may be used to monitor mmServer and job processing status on the local host or on remote hosts.

mmVisor configuration file

The mmVisor executable files are located in the %MM_HOME%\bin folder. There is a configuration file, mmVisor.usr.xml, for each user. mmVisor.usr.xml files are located in each user's %HOMEPATH%\.mmVisor folder.

Starting mmVisor

The mmVisor interface is started by activating the mmVisor desktop icon, illustrated in Figure 5.1.

Figure 5.1 mmVisor desktop icon

By default mmVisor attempts to connect to mmServer on the local host. If mmServer is not running on the local host mmVisor will display empty fields. If mmServer is running on a remote host the contact details of that host must be entered by the user. See Connecting to a remote mmServer host on page MS-36.

Starting the mmServer service

The mmServer service must be running before any jobs can be processed. The mmServer service may be started by clicking the Start button in the Control tab.

The mmServer service may fail to start owing to a problem with the FrameMaker installation or configuration. Pressing the key combination Ctl-Alt-f in the mmVisor window will start a FrameMaker session as the RunAs user and will display associated configuration errors and alerts.

Starting mmServer via mmVisor also starts the mmConnect remote monitoring and job processing secure sockets interface (see pages MS-19–22).

Monitoring mmServer and job status (Status tab)

.

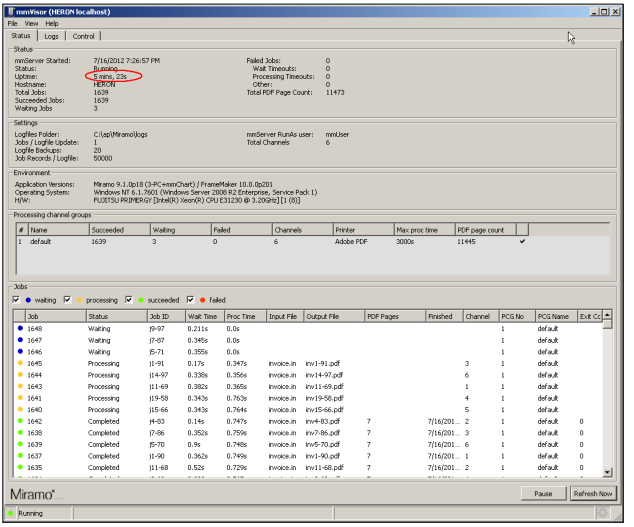

Figure 5.2 mmVisor Status, Settings, Environment and jobRecord screen

When the mmVisor desktop icon is activated the result is a screen display similar to that illustrated in Figure 5.2.

The default view shown in Figure 5.2 may be changed by clicking on View and deselecting some or all of the Server Settings, Server Status, Environment and Processing Channel Groups view options. This will expose more space for displaying jobRecord data, as illustrated in Figure 5.3. A maximum of 200 jobRecords can be displayed in the Status tab.

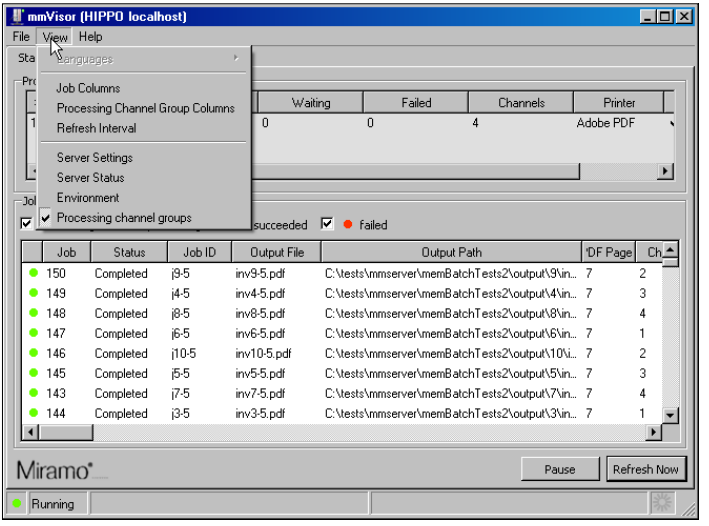

Figure 5.3 Displaying more job records

Viewing log file job records (Logs tab)

Log file job records may be viewed by clicking on the Logs tab. The first time this tab is clicked during a mmVisor session the data fields are empty, as shown in Figure 5.4. To view a logfile the Select Log File button must be clicked to activate the logfile selector (Figure 5.5).

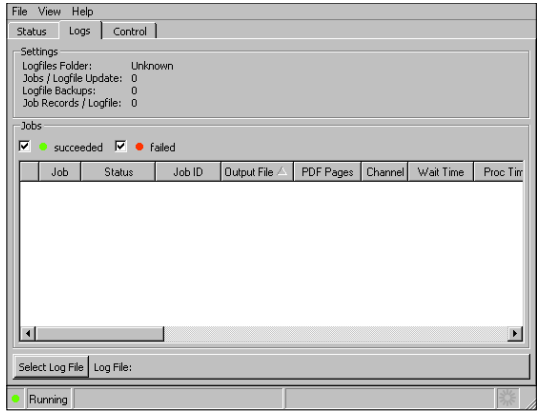

Figure 5.4 Log File viewer, no logfile selected

The logfile selector displays a list of all the mmServer logfiles in the logfiles folder. By default logfiles are located in the folder: %MM_HOME%\logs. The logfile folder may be changed using the Log Folder text box in the mmVisor control tab. By default a maximum of 20 backup logfiles are preserved, each containing a default maximum of 50 000 job records. These defaults may be changed using the Log Folder Advanced button on the

control tab. See Figure 5.8 on MS-35.

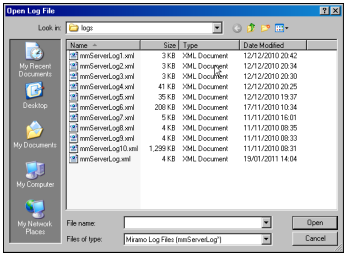

Figure 5.5
Logfile selector

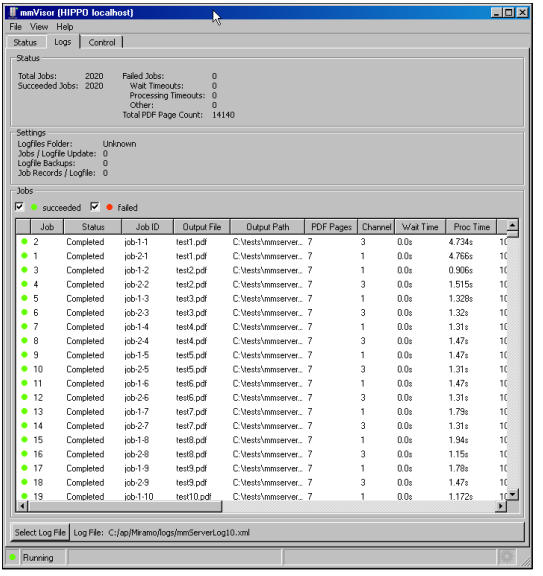

Figure 5.6 Log file jobRecord data display

The mmServer operating parameters in effect when the logfile was created and job record data for all jobs in the selected logfile are then displayed, as illustrated in Figure 5.5. A new logfile is started, and backup logfiles are cycled, every time mmServer is started.

mmVisor cannot open mmServer logfiles on a remote host. To view logfiles from a remote host, they must be copied from the remote host to the mmVisor host.

mmServer configuration and control (Control tab)

The Control tab enables stopping and starting the mmServer service and changing its operating parameters. The initial view of the Control tab is shown in Figure 5.7.

The following are the main operating parameters that may be changed.

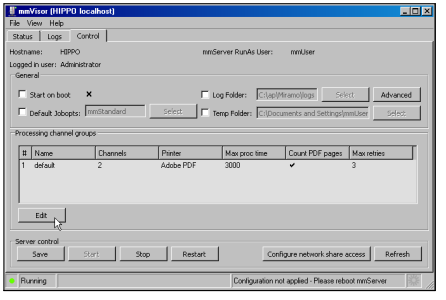

Figure 5.7 Initial view of Control tab

- **Start on boot** By default the mmServer service is set to start automatically whenever the host computer is rebooted. Clicking on Start on boot enables switching the default behavior to require manual starting of the mmServer service.

- **User mode** By default all users are able to run Miramo jobs. Clicking on User mode enables restricting jobs to being runable by the runAs user only. Note that all jobs are *always* run by the runAs user only, i.e. the default runAs user, .\mmuser, in the case illustrated in Figure 5.7 (see towards upper right corner).

- **Log Folder** By default log files are created in the %MM_HOME%\logs folder. The location for log files may be changed by clicking on Log Folder.

- **Temp folder** Multiple temporary files are created while a job is being processed. By default these temporary files are created in the folder %TMP%, the same as the %TEMP% folder, for the runAs user. The folder for temporary files used by FrameMaker and Acrobat distiller may be changed by clicking on Temp folder
- **Configure network share access** Enable the mmServer Run-As user to access network folder resources.

Configuring processing channel groups (Control tab)

A processing channel group constitutes a self-contained, distinct mmServer job queue. A processing channel group may be configured with from one to a maximum of ten processing channels. A maximum of ten processing channel groups may be configured. The aggregate total of all processing channels cannot exceed ten.[1] Jobs are assigned to a processing channel group using the '-pcg', or the '-processGroup', command line option. See page R-7 in the *Miramo Reference Guide*), or by using a corresponding API function.

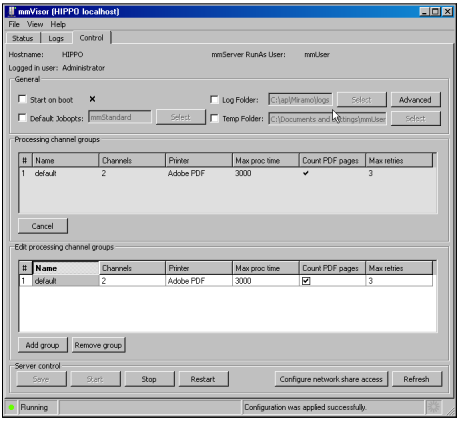

Figure 5.8 Configuring processing channel groups (1)

The first step in configuring processing chanel groups is to click on Edit (Figure 5.7). This exposes the processing channel group configuration panel, as illustrated in Figure 5.8.

A new processing channel group is added by clicking on the Add group button, as illustrated in Figure 5.9.

The default name for processing channel group 1 is 'default'. Additional processing channel groups are named 'new', 'new1', 'new2' and so on. These names, along with the values of other processing channel group properties, can be changed by clicking in the value field, typing a new value and pressing Enter.

After setup and configuration of processing channel groups has been completed the new configuration must be save by

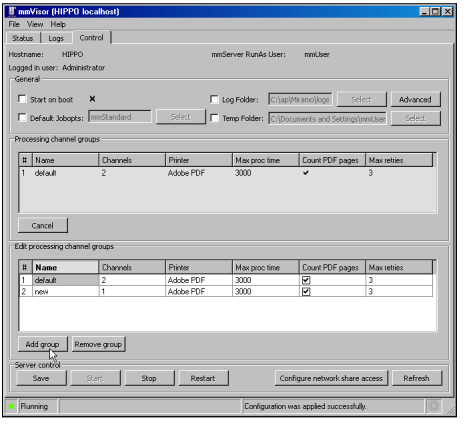

Figure 5.9 Configuring processing channel groups (2)

1. These figures relate to installations with an unlimited processing channel license. Different license versions restrict the total number of processing channels. Additional practical restrictions apply depending on the number of CPU cores and RAM in the host machine.

clicking the Save button (see Figure 5.10). The new configuration will take effect only when mmServer is next started, or stopped and re-started. Jobs that are run without using the '-pcg', or the '-processGroup', command line option (page R-7) or by using the corresponding API function, are always assigned to processing channel group 1, regardless of its name.

A processing channel group may be deleted by selecting its number and clicking Remove group.

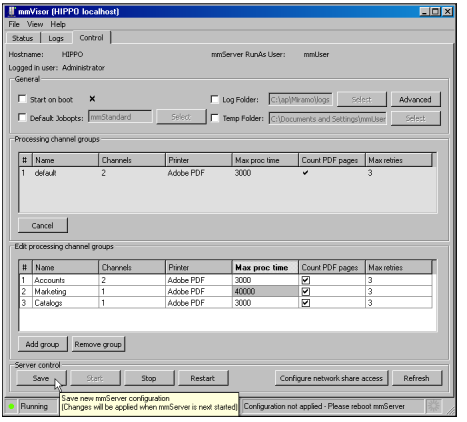

Figure 5.10 Configuring processing channel groups (3)

Viewing server status and job status on a remote host

mmVisor can view Miramo status and job status on a remote host, provided both mmServer and mmConnect are running on the remote host. Logfiles cannot be viewed on a remote host. Input and output files cannot be viewed on a remote host. Running and waiting jobs may be terminated on a remote host.

Connecting to a remote mmServer host

To access a remote host running mmServer and mmConnect select File then select Connect, as illustrated in Figure 5.11. If the name or IP address of the remote host appears in the list of Servers, illustrated in Figure 5.12, double-click on the entry. If the server is not listed, click on Add and enter the details of the remote host in the Configuration panel.

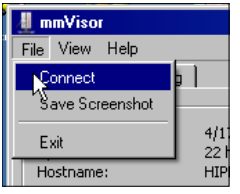

Figure 5.11 File menu, Connect

Clicking on Add (Figure 5.12) displays the Server Details panel shown in Figure 5.13. Four data fields need to be completed to set up a connection to a remote mmServer host.

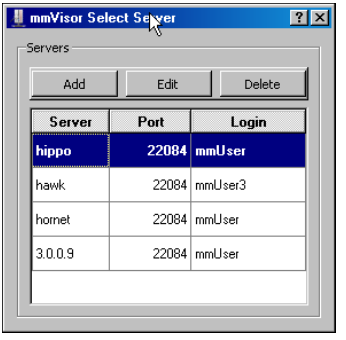

Figure 5.12 Select Server menu

– *mmServer host name:* The IP address or node name of the remote host running mmServer.

– *mmConnect port number:* The mmConnect port number on the remote host set by the mmConnect '-port' option when mmConnect is started on the remote host (see page MS-21). The value for the mmConnect port number can be checked by running the command

  mmConnect -status

  on the remote host. The default value is

22084.

– *mmServer RunAs user:* The mmServer RunAs user name on the remote host, either as assigned during installation or subsequently changed using the mmServer '-runAs' option (see page MS-6.) The name of the RunAs user can be checked by running the command

  mmserver -status ras

  on the remote host. A preceding '.\' in the value of the runAs attribute in the output from the above command should be ignored. The default value is mmUser.

– *RunAs user password:* The mmServer RunAs user password on the remote either as assigned during installation or subsequently changed.

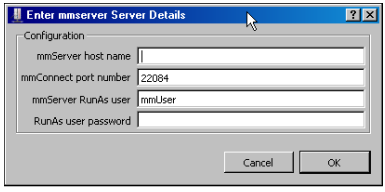

Figure 5.13 Connection Details (mmServer host)

After initial configuration remote host connection details may be modified by selecting the host in the Select Servers panel and clicking on the Edit button.

Setting the refresh interval

The mmVisor data update frequency is specified by setting the refresh interval. Click on the View button, then on Refresh Interval (Figure 5.14). This displays the slider shown in Figure 5.15.

Figure 5.14 View Menu, Refresh Interval

The slider shown in Figure 5.15 enables the interval between data updates to be set in the range from 1 to 60 seconds. Lower values in this range result in a slight increase in system load. Updating may be suspended by selecting the Pause button (Figure 5.2 on page MS-32) or by minimizing or terminating all mmVisor application instances. The default refresh interval is five seconds.

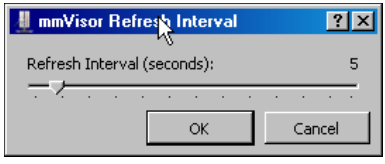

Figure 5.15 set Refresh Rate

Selecting job data columns

Job data columns may be selected or deselected for viewing.

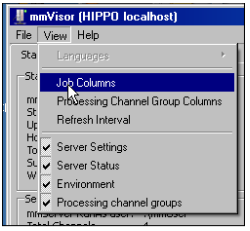

Figure 5.16 View Menu, Columns

Select View, then Columns, as shown in Figure 5.16.

The list of selectable columns is shown in Figure 5.17. The column widths may be adjusted by placing the cursor between the column headings (see Figure 5.2 on page MS-32) and moving it left or right. The column selection and the column widths of the last mmVisor session to be closed is remembered between mmVisor sessions.

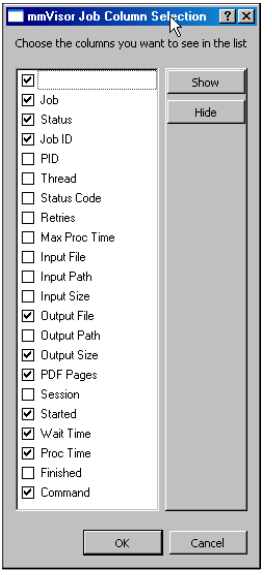

Figure 5.17 Job Column Selection

Viewing input and output files

Input files for waiting, processing and completed jobs may be viewed by selecting a job record and right-clicking and then selecting Open Input File, as illustrated in Figure 5.18. This applies to current jobs and when viewing logfiles.

Output files from completed jobs that are in PDF format may be viewed in the same way by selecting Open Input File in place of Open Output File.

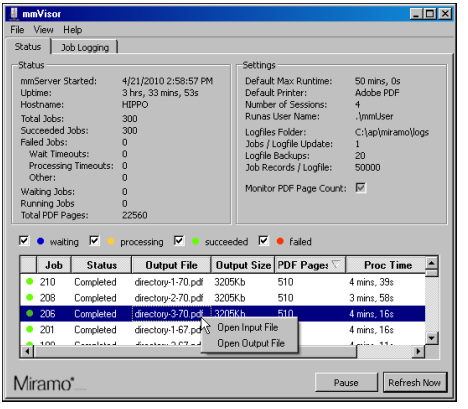

Figure 5.18 Viewing input and output files

Terminating waiting and running jobs

Waiting and running (Processing) jobs may be terminated by selecting a job record and right-clicking and then selecting Terminate Job, as illustrated in Figure 5.19.

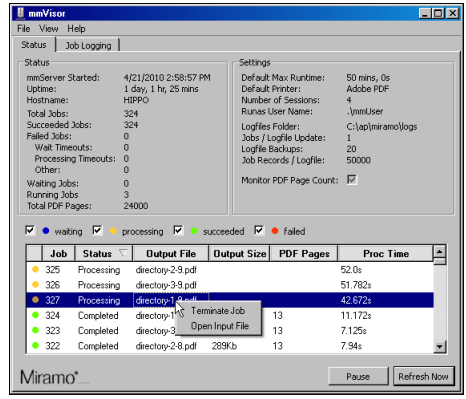

Figure 5.19 Terminating jobs