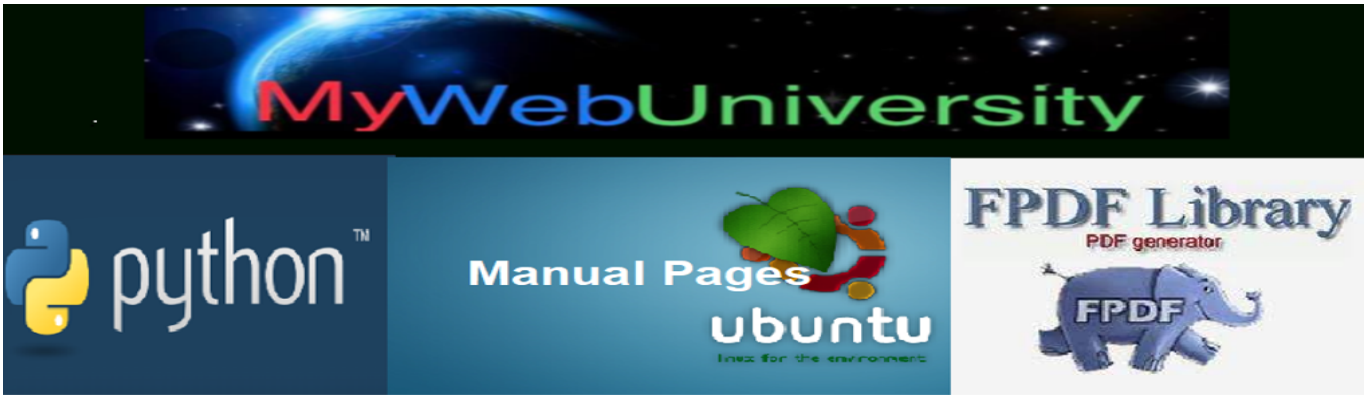

Rocky Enterprise Linux 9.2 Manual Pages on command 'jfr.1'

$ man jfr.1

JFR(1) JDK Commands JFR(1)

NAME

jfr - print and manipulate Flight Recorder files

SYNOPSIS

To print the contents of a flight recording to standard out:

jfr print [options] file

To display aggregated event data on standard out:

jfr view [options] file

To configure a .jfc settings file:

jfr configure [options]

To print metadata information about flight recording events:

jfr metadata [file]

To view the summary statistics for a flight recording file:

jfr summary file

To remove events from a flight recording file:

jfr scrub [options] file

To assemble chunk files into a flight recording file:

jfr assemble repository file

To disassemble a flight recording file into chunk files:

jfr disassmble [options] file

options

Optional: Specifies command-line options separated by spaces. See the individual subcomponent sections for descriptions of the available options.

file Specifies the name of the target flight recording file (.jfr).

repository

Specifies the location of the chunk files which are to be assembled into a flight recording.

DESCRIPTION

The jfr command provides a tool for interacting with flight recorder files (.jfr). The main function is to filter, summarize and output flight recording files into human read? able format. There is also support for scrubbing, merging and splitting recording files. Flight recording files are created and saved as binary formatted files. Having a tool that can extract the contents from a flight recording and manipulate the contents and translate them into human readable format helps developers to debug performance issues with Java applications.

Subcommands

The jfr command has several subcommands:

? print

? view

? configure

? metadata

? summary

? scrub

? assemble

? disassemble

jfr print subcommand

Use jfr print to print the contents of a flight recording file to standard out.

The syntax is:

jfr print [--xml|--json] [--categories <filters>] [--events <filters>] [--stack-depth <depth>] <file>

where:

--xml Print the recording in XML format.

--json Print the recording in JSON format.

--categories <filters>

Select events matching a category name. The filter is a comma-separated list of names, simple and/or qualified, and/or quoted glob patterns.

--events <filters>

Select events matching an event name. The filter is a comma-separated list of names, simple and/or qualified, and/or quoted glob patterns.

--stack-depth <depth>

Number of frames in stack traces, by default 5.

<file> Location of the recording file (.jfr)

The default format for printing the contents of the flight recording file is human read? able form unless either xml or json is specified. These options provide machine-readable output that can be further parsed or processed by user created scripts.

Use jfr --help print to see example usage of filters.

To reduce the amount of data displayed, it is possible to filter out events or categories of events. The filter operates on the symbolic name of an event, set by using the @Name annotation, or the category name, set by using the @Category annotation. If multiple fil? ters are used, events from both filters will be included. If no filter is used, all the events will be printed. If a combination of a category filter and event filter is used, the selected events will be the union of the two filters.

For example, to show all GC events and the CPULoad event, the following command could be used:

jfr print --categories GC --events CPULoad recording.jfr

Event values are formatted according to the content types that are being used. For exam? ple, a field with the jdk.jfr.Percentage annotation that has the value 0.52 is formatted as 52%.

Stack traces are by default truncated to 5 frames, but the number can be increased/de? creased using the --stack-depth command-line option.

jfr view subcommand

Use jfr view to aggregate and display event data on standard out.

The syntax is:

jfr view [--verbose] [--width <integer>] [--truncate <mode>] [--cell-height <integer>] <view> <file>

where:

--verbose

Displays the query that makes up the view.

--width <integer>

The width of the view in characters. Default value depends on the view.

--truncate <mode>

How to truncate content that exceeds space in a table cell. Mode can be 'begin?

ning' or 'end'. Default value is 'end'.

--cell-height <integer>

Maximum number of rows in a table cell. Default value depends on the view.

<view> Name of the view or event type to display. Use jfr --help view to see a list of

available views.

<file> Location of the recording file (.jfr)

The <view> parameter can be an event type name. Use the jfr view types <file> to see a

list. To display all views, use jfr view all-views <file>. To display all events, use

jfr view all-events <file>.

jfr configure subcommand

Use jfr configure to configure a .jfc settings file.

The syntax is:

jfr configure [--interactive] [--verbose] [--input] [--output] [option=value]* [event-

setting=value]*

--interactive

Interactive mode where the configuration is determined by a set of questions.

--verbose

Displays the modified settings.

--input <files>

A comma-separated list of .jfc files from which the new configuration is based. If

no file is specified, the default file in the JDK is used (default.jfc). If 'none'

is specified, the new configuration starts empty.

--output <file>

The filename of the generated output file. If not specified, the filename cus?

tom.jfc will be used.

option=value

The option value to modify. To see available options, use jfr help configure

event-setting=value

The event setting value to modify. Use the form: <event-name>#<setting-name>=<val?

ue> To add a new event setting, prefix the event name with '+'.

The whitespace delimiter can be omitted for timespan values, i.e. 20ms. For more infor?

mation about the settings syntax, see Javadoc of the jdk.jfr package.

jfr metadata subcommand

Use jfr metadata to display information about events, such as event names, categories and

field layout within a flight recording file.

The syntax is:

jfr metadata [--categories] [--events] []

--categories <filter>

Select events matching a category name. The filter is a comma-separated list of

names, simple and/or qualified, and/or quoted glob patterns.

--events <filter>

Select events matching an event name. The filter is a comma-separated list of

names, simple and/or qualified, and/or quoted glob patterns.

<file> Location of the recording file (.jfr)

If the parameter is omitted, metadata from the JDK where the 'jfr' tool is located will be

used.

jfr summary subcommand

Use jfr summary to print statistics for a recording. For example, a summary can illus?

trate the number of recorded events and how much disk space they used. This is useful for

troubleshooting and understanding the impact of event settings.

The syntax is:

jfr summary <file>

where:

<file> Location of the flight recording file (.jfr)

jfr scrub subcommand

Use jfr scrub to remove sensitive contents from a file or to reduce its size.

The syntax is:

jfr scrub [--include-events <filter>] [--exclude-events <filter>] [--include-categories

<filter>] [--exclude-categories <filter>] [--include-threads <filter>] [--exclude-threads

<filter>] <input-file> [<output-file>]

--include-events <filter>

Select events matching an event name.

--exclude-events <filter>

Exclude events matching an event name.

--include-categories <filter>

Select events matching a category name.

--exclude-categories <filter>

Exclude events matching a category name.

--include-threads <filter>

Select events matching a thread name.

--exclude-threads <filter>

Exclude events matching a thread name.

<input-file>

The input file to read events from.

<output-file>

The output file to write filter events to. If no file is specified, it will be written to the same path as the input file, but with "-scrubbed" appended to the filename.

The filter is a comma-separated list of names, simple and/or qualified, and/or quoted glob patterns. If multiple filters are used, they are applied in the specified order.

jfr assemble subcommand

Use jfr assemble to assemble chunk files into a recording file.

The syntax is:

jfr assemble <repository> <file>

where:

<repository>

Directory where the repository containing chunk files is located.

<file> Location of the flight recording file (.jfr).

Flight recording information is written in chunks. A chunk contains all of the informa? tion necessary for parsing. A chunk typically contains events useful for troubleshooting. If a JVM should crash, these chunks can be recovered and used to create a flight recording file using this jfr assemble command. These chunk files are concatenated in chronological order and chunk files that are not finished (.part) are excluded.

jfr disassemble subcommand

Use jfr disassemble to decompose a flight recording file into its chunk file pieces.

The syntax is:

jfr disassemble [--max-chunks <chunks>] [--output <directory>] <file>

where:

--output <directory>

The location to write the disassembled file, by default the current directory

--max-chunks <chunks>

Maximum number of chunks per file, by default 5. The chunk size varies, but is typically around 15 MB.

--max-size <size>

Maximum number of bytes per file.

<file> Location of the flight recording file (.jfr)

This function can be useful for repairing a broken file by removing the faulty chunk. It can also be used to reduce the size of a file that is too large to transfer. The result? ing chunk files are named myfile_1.jfr, myfile_2.jfr, etc. If needed, the resulting file names will be padded with zeros to preserve chronological order. For example, the chunk file name is myfile_001.jfr if the recording consists of more than 100 chunks.

jfr version and help subcommands

Use jfr --version or jfr version to view the version string information for this jfr com? mand.

To get help on any of the jfr subcommands, use:

jfr <--help|help> [subcommand]

where:

[subcommand] is any of:

? print

? view

? configure

? metadata

? summary

? scrub

? assemble

? disassemble