



Deltek

Deltek Maconomy®

Application Performance Monitoring
(APM)

September 20, 2019

While Deltek has attempted to verify that the information in this document is accurate and complete, some typographical or technical errors may exist. The recipient of this document is solely responsible for all decisions relating to or use of the information provided herein.

The information contained in this publication is effective as of the publication date below and is subject to change without notice.

This publication contains proprietary information that is protected by copyright. All rights are reserved. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, or translated into another language, without the prior written consent of Deltek, Inc.

This edition published September 2019.

© Deltek, Inc.

Deltek's software is also protected by copyright law and constitutes valuable confidential and proprietary information of Deltek, Inc. and its licensors. The Deltek software, and all related documentation, is provided for use only in accordance with the terms of the license agreement. Unauthorized reproduction or distribution of the program or any portion thereof could result in severe civil or criminal penalties.

All trademarks are the property of their respective owners.

Contents

Overview	3
Log Modes.....	4
Log Levels	5
Log Entries	6
Configuration	8
server.ini	8
logback.xml.....	10
Log Events	11
Overview.....	11
Authentication	14
Service Provider Requests	15
Workspace Client Requests	15
WebService Requests	17
WebDaemon Requests	17
Container Requests.....	18
Container Contributions.....	22
Remote Procedure Calls	23
MSL Requests	23
Database Requests	25
Security Log.....	25
Callbacks	25
Notifications	26
Sample Queries	27
Request Execution Time	27
Workspace Requests	28
WebService and WebDaemon Requests.....	28
Login Auditing	28
Active Request	29

Overview

The Application Performance Monitor (APM) is a new logging framework introduced in Maconomy 2.3.4 in a partial form, and in Maconomy 2.3.5/2.4.4 as a complete release.

The APM framework is designed to serve several different monitoring purposes:

1. Audit logging of all user logins
2. High-level activity logging of incoming requests
3. Detailed drill-down logging of long-running requests

The APM framework consists of a core logging framework, and a number of loggers distributed throughout the Maconomy technology stack. When triggered during the execution of a request, these loggers will create log events that measure the time spent of a specific aspect of the request execution.

The generated log entries are passed to the core logging framework, where they are filtered according to the framework configuration settings. The purpose of this filtering is to ensure that only log entries which are of interest are included in the final log output.

In addition to tracking the performance of the workspace and container contribution layers in the Coupling Service, the APM framework can also collect log events from a number of Maconomy Server-side loggers, from the Maconomy Application layer and all the way down to the database interface.

This means that it is possible to quantify what fraction of a long-running request is spent on application processing and database queries. Combined with the high-level information provided by the workspace and/or container contributions involved in the request, this can provide a great starting point for a targeted effort to identify and improve on specific performance bottlenecks anywhere in the Maconomy technology stack.

Log Modes

The APM framework can operate in two modes:

Direct: Log entries are written directly to the SLF4J logger. This is the most verbose logging mode and can be expected to produce large quantities of log entries on a production system.

Delayed: Log entries are queued with a configurable delay. If the total duration of a log event is smaller than the delay its entries will not be logged. This is the recommended mode.

The delayed log mode is enabled by configuring a minimum time duration in the monitor configuration in 'configuration/server.ini'. Please refer to the section Configuration below for more information.

Entries with log level 'ERROR' and entries marked with the 'Audit' flag are always logged.

Log Levels

All log events are associated with an SLF4J log level, which is applied to the log entries generated by the event. The monitor logger declared in the Logback configuration will compare this level with its own level to determine if the log entry should be included in the log output. The default log level is INFO, but a few log events set a lower level such as DEBUG. If the operation monitored by a log event trigger a serious or unexpected error, the level of all subsequent log entries from that log even will be raised to ERROR.

A log entry can declare a number of key-value pairs containing extra information about the event being logged. Each of these key-value pairs are also assigned a log level. Less verbose values are generally assigned a high level (typically INFO), while more verbose values might be given a lower level (DEBUG or TRACE).

By changing the log level of the Logback monitor logger it is possible to control both the number of log entries included in the log output, and to some extent the verbosity of each log entry:

ERROR: Only monitor serious error conditions occurring while serving requests.

INFO: Used for high-level activity monitoring. This will provide an overview of system activity while skipping some of the more verbose fields, such as key field information and search restriction on container requests.

DEBUG: Provides more detailed monitoring of system performance. This will include key field information and search restrictions for contain requests, which will be helpful when investigating the runtime complexity of the different requests.

TRACE: All log events and all key-values. The additional information included at this level is rarely needed, and it is generally not a recommended log level to use in production.

Regardless of which log level is chosen it is recommended to always configure the APM framework with a minimum time duration, to reduce the volume of log entries included in the log output. Please see the section Configuration below for more information.

Log Entries

Log events created by the APM framework follow a simple life cycle:

Enter: The log event is created and an 'Enter' message is issued. (optional)

Message: Additional 'Message' entries issued for an active log event. (optional)

Exit: The log event is closed and an 'Exit' message is issued.

Each log entry produced by a log event consists of a collection of structured key-value pairs (Key="Value"). This format - while somewhat verbose – is quite readable, but it is primarily designed to be easily consumable by external log monitoring frameworks such as Splunk or Logstash.

The information included in each log entry is designed with reporting in mind, and therefore contains several ID fields which group the entries on different dimensions, such as those belonging to the same request, or to the same user login session.

The APM framework can be configured with various time duration thresholds used to omit log entries which are considered too fast to be of interest. For this reason the Enter and Message entries of a log event might fall below the configured threshold and be omitted. If the entire event falls below the threshold the Exit entry will also be omitted, unless another condition dictates that it should be included anyway.

Each log entry is annotated with the following information:

Field	Description
Time	A timestamp for when the log entry was created (<i>not</i> when it was written).
LogDepth	The nesting level of the log entry.
Thread	The name of the thread that created the log entry.
Name	A standardized name for the logger.
Type	Either 'Enter', 'Message' or 'Exit'.
Level	An SLF4J log level for the log entry. If this level is lower than the log level of the Logback logger ('com.maconomy.util.logger.McMonitorLogger') the log entry will not be written.
Duration Elapsed	The time duration (in milliseconds) since the log event was created. If the time measurement is not entirely contained within the scope of the log event then it will be reported as 'Elapsed' rather than 'Duration'.
Values*	Additional key-value pairs. Each value is annotated with an SLF4J log level for controlling whether the value should be included in the log entry.
Error	Any error that was encountered while executing the logged action (optional). If the error is considered serious (internal/unexpected) the log entry will be elevated to ERROR level, otherwise it will remain at its current level.
Audit	A flag ('true/false') indicating whether the log entry contains audit-related information. (optional, 'false' if absent)

Log Entries

PrincipalName	A Maconomy User- or System Principal Name describing the available user and/or system information (e.g. macoprod.en_US\Administrator\Standard).
EntryID	A semi-unique ID identifying log lines generated by the same log entry.
RequestID	A unique ID identifying the request that the log entry belongs to. (optional)
ContextID	A unique ID identifying the wider context that the request appears in - e.g. the user's login session. (optional)
Connection	<p>A description of the connection to the client. (optional, introduced in: 2.3.5, 2.4.4)</p> <p>If present the description can be expected to adhere to the following format:</p> <p><code>[obj[@id]] l(addr.port)<->r(addr.port)</code></p> <p>-where:</p> <p>'obj' is the object providing the connection information</p> <p>'id' is the instance ID of 'obj' (Java Identity Hash Code)</p> <p>'l(...)<->r(...)' describe the local<->remote connection endpoints</p> <p>'addr' is the endpoint IP address</p> <p>'port' is the endpoint port number</p>

Configuration

The APM framework is configured in 'configuration/server.ini' and 'configuration/logback.xml' on the Coupling Service. The default versions of these files contain a suggested setup with explanatory comments and sensible default values.

All settings are dynamically monitored, and can be changed on the fly without having to restart the Coupling Service or any other Maconomy component.

server.ini

The table below describes all available settings in 'server.ini':

Setting	Description	Default
log.monitor.*		
enabled	Change to 'true' to enable monitor logging.	false
time-unit	The time unit for 'minimum-time'. Must be a unique prefix of 'milliseconds', 'seconds' or 'minutes'.	seconds
minimum-time	The minimum time duration for log entries to be printed. Setting this to '0' will put the framework in 'direct' mode.	0
minimum-exit-time	The minimum time duration for 'Exit' log entries to be printed. This can only be set to a value <= 'minimum-time'	<i>minimum-time</i>
time-format	The date/time format pattern to use for the time stamp on each log line.	yyyy-MM-dd HH:mm:ss.SSSZ
mdc-keys	A comma-separated list of keys for the Mapped Diagnostic Context (MDC) values to include in all log entries.	n/a
include, exclude	Comma-separated lists with string patterns for log entry names to explicitly include/exclude. <ul style="list-style-type: none"> If only an include list is specified then log entries will only be included if they match one of the include patterns. If only an exclude list is specified then log entries will be included unless they match one of the exclude patterns. If both an include and an exclude list is specified then the include list contains exemptions from the exclude list. <p>I.e. if the log entry name matches one of the patterns in the exclude list then it will only be included if it also matches one of the patterns in the include list.</p>	n/a
audit	A comma-separated list of string patterns for log entry names that are auditable, and which should therefore be logged even if their duration is below the minimum time duration.	n/a
exit-log-depth	Deprecated (use 'audit' instead) Write exit messages for the first N log entry levels, even if they don't meet the minimum time duration criteria. (only in: 2.4.4)	1

log.monitor.server.*		
enabled	Explicitly enable/disable server-side monitor logging.	<i>log.monitor.enabled</i>
include	A comma-separated list of the server-side loggers to enable.	<i>n/a</i>
all-entries	Bypass the "delay queue" in each server process and send all server-side log entries to the Coupling Service. When this option is not used a default minimum delay is configured on each server process, to conserve network bandwidth and reduce log entry volume. At the time of writing this minimum delay is set to 10 milliseconds.	false
push	Use push messages to send server-side log entries to the Coupling Service	true

Below is shown an example of how the APM framework could be configured:

```
log.monitor {
    enabled = true
    time-unit = seconds

    # The minimum time duration for log entries to be printed.
    minimum-time = 30

    # The minimum time duration for 'Exit' log entries to be printed.
    minimum-exit-time = 5

    # The maximum length of values including in log entries.
    maximum-value-length = 200

    # A comma-separated list of string patterns matching the names of log
    # entries that are auditable, and which should therefore be logged
    # even if their durations are below the minimum time duration.
    audit = Workspace:*, WebService:*, WebDaemon:*

    # A comma-separated list of the server-side loggers to enable.
    server.include = Database,SecurityLog,WebComm,MScript,MSL,RPC
}
```

In this example the APM framework is configured to provide:

- Top-level monitoring of all Workspace Client-, Web- and WebDaemon requests
- Drill-down logging of call trees exceeding 5 seconds
- Additional Enter/Message logging of requests exceeding 30 seconds

The 'server.include' setting lists all the server-side loggers which are available for monitoring.

logback.xml

All log entries generated by the APM framework are produced by a single SLF4J logger in the Coupling Service. The standard Logback configuration file ('configuration/logback.xml') is used to direct these log entries to an output file (by default 'log/coupling/maconomy-monitor.log').

Below is shown an example of how this could be configured:

```
<appender name="MONITOR" class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>
    ${logback.output.directory}/maconomy-monitor.log
  </file>
  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <fileNamePattern>
      ${logback.output.directory}/maconomy-monitor-%d{yyyy-MM-dd}.log
    </fileNamePattern>
    <maxHistory>30</maxHistory>
  </rollingPolicy>
  <layout class="ch.qos.logback.classic.PatternLayout">
    <Pattern>%msg%n</Pattern>
  </layout>
</appender>

<logger name="com.maconomy.util.logger.McMonitorLogger" additivity="false">
  <level value="INFO" />
  <appender-ref ref="MONITOR" />
</logger>
```

In this example the threshold log level is set to INFO. Please refer to the section Log Levels above to learn more about the implications of setting this level.

Log Events

The following sections describe the different types of log events that are generated by the APM framework.

With a few exceptions log events with names on the form 'MaconomyServer:LoggerName' are generated by the Maconomy Server processes, and require the logger name after the ':' to be included in the 'log.monitor.server.include' setting. (see Configuration above)

Two exceptions to this rule are the 'MaconomyServer:Request' and 'MaconomyServer:Callback' events which are produced by the Coupling Service before calling remote functions and when receiving callbacks from the Maconomy Server process.

Overview

This section provides an overview of all log events included in the APM framework.

Authentication	Audit	LoginRules	ApplyAll
Auth:Login	•	•	•
Auth:Logout	•	•	
Auth:Context			
Service Provider Request	Message		
ServiceProvider:GetLease	•		
Workspace Client Requests	WorkspaceName	PaneRequest	PaneNames
Workspace:Spec	•	•	•
Workspace:PaneSpec	•	•	•
Workspace:Data	•	•	•

Log Events

WebService / WebDaemon Requests	Received / Sent									
	RemoteAddr	Request	Query	Url	Method	RestrictionData	ForeignKeyName	RestoreData	TargetRestriction	SourceRestriction
WebService:Request					
WebDaemon:Request					
Container Requests										
	ActionName	NewValues	Parameters	Restriction	ContainerName	Operation	SourceRestriction	TargetRestriction	RestoreData	ForeignKeyName
Container:Open					.					
Container:Close					.					
Container:Specify					.					
Container:Lock (2.3.4 only)						
Container:Unlock (2.3.4 only)						
Container:Initialize					
Container:Create					
Container:Read					
Container:Update					
Container>Delete					
Container:Action				
Container:Print					
Container:Move
Container:Restore
Container:Restrict

Log Events

Container Contributions	ContainerName	EventId	ContributionId	BundleId	PhaseId	ActionName
Container:Contribution	•	•	•	•	•	•
Remote Procedure Calls	Function	Parameters	PID			
MaconomyServer:Request	•	•	•			
MaconomyServer:Callback	•	•	•			
Maconomy Server Requests	Audit	Context	PID			
MaconomyServer:RPC		•	•			
MaconomyServer:WebComm		•	•			
MaconomyServer:MScript		•	•			
MaconomyServer:MSL		•	•			
MaconomyServer:Database		•	•			
MaconomyServer:SecurityLog	•	•	•			

Authentication

The following authentication related log events exist:

Name	Description	Since
Auth:Login	Issued when login rules are being (re-)applied for a user.	2.3.4, 2.4.4
Key	Value	Log Level
LoginRules	The login rules being applied	INFO
ApplyAll	Whether the login rules are applied unconditionally	INFO
Audit	Set to 'true' on the 'Exit' entry to classify it as audit-related (since 2.3.5)	n/a

Name	Description	Since
Auth:Logout	Issued when a user is logged out of all previously applied login	2.3.4, 2.4.4
Key	Value	Log Level
LoginRules	The login rules that were applied	INFO
Audit	Set to 'true' on the 'Exit' entry to classify it as audit-related (since 2.3.5)	n/a

Name	Description	Since
Auth:Context	Issued when an authentication context is established for an existing user session. These log events track the total execution time of each action the user performs, including the administrative overhead (e.g. acquire/release a server instance, reconnect/disconnect user, etc.)	2.3.4, 2.4.4
Key	Value	Log Level
	<i>No additional key-values</i>	

Service Provider Requests

The Service Provider Request event tracks the time spent obtaining a MaconomyServer service lease from the service pool:

Name	Description	Since
ServiceProvider:GetLease	Issued when a service lease is requested from a service pool. Usually these requests will be very fast and therefore filtered away, but if the service pool is running full or is otherwise unable to scale to meet demand then requests will be queued while waiting for a service to become available.	2.3.5, 2.4.4
Key	Value	Log Level
Message	The service being requested: 'Service: <i>service-description</i> '	INFO

Workspace Client Requests

The following log events are issued for Workspace Client Requests:

Name	Description	Since
Workspace:Spec	Issued when workspace specification request is performed.	2.3.4, 2.4.4
Key	Value	Log Level
WorkspaceName	The name of the workspace	INFO
PaneRequest	A descriptive text for the pane request part of the spec request	INFO
PaneNames	A list with all pane names in the workspace request tree	INFO

Name	Description	Since
Workspace:PaneSpec	Issued when a workspace pane specification request is performed, i.e. a request to attach additional panes to a workspace - typically Foreign-Key search panes.	2.3.4, 2.4.4
Key	Value	Log Level
WorkspaceName	The name of the workspace	INFO
PaneRequest	A descriptive text for the pane request part of the pane spec request	INFO
PaneNames	A list with all pane names in the workspace request tree	INFO

Log Events

Name	Description	Since
Workspace:Data	Issued when a workspace data request is performed.	2.3.4, 2.4.4
Key	Value	Log Level
WorkspaceName	The name of the workspace	INFO
PaneRequest	A descriptive text for the pane request part of the pane spec request	INFO
PaneNames	A list with all pane names in the workspace request tree	INFO

WebService Requests

The following top-level log entries are issued for the REST-full web service requests:

Name	Description	Since
WebService:Request	Issued when a web service request is performed.	2.3.5, 2.4.4
Key	Value	Log Level
Method	The HTTP method of the request (GET/POST/...)	INFO
Url	The complete URL of the request (protocol://server:port/path), but not including the query string	INFO
Query	The query string of the request	INFO

WebDaemon Requests

The following top-level log entries are issued when a WebDaemon request is performed:

Name	Description	Since
WebDaemon:Request	Issued when a WebDaemon request is performed (Java client, Analyzer or MScript). WebDaemon requests pass through the Coupling Service as a binary protocol, but they can trigger Maconomy Server generated log events.	2.3.5, 2.4.4
Key	Value	Log Level
RemoteAddr	The value of the X_FORWARDED_FOR or REMOTE_ADDR CGI environment variable passed to the CGI program (optional)	INFO
Method	The value of the REQUEST_METHOD CGI environment variable passed to the CGI program (optional)	INFO
Url	The invoked CGI request URL (without query parameters) OR the CGI executable path and script name (for command-line invocations)	INFO
Query	The query parameters passed to the CGI program, either via the QUERY_STRING CGI environment variable, or on the command-line (for command-line invocations). This field will only be present for requests where the CGI client has not supplied a request description ('Request').	INFO
Request	A short description of the request. If a request description is not supply by the CGI client the default value will be the value of the PATH_INFO CGI environment variable	INFO
Received	The number of bytes received from the CGI program	INFO
Sent	The number of bytes sent to the CGI program	INFO

Log Events

PrincipalName	Deprecated (replaced by an MDC provided 'PrincipalName'): The system principal of the server pool that will execute the request (only in: 2.3.4)	INFO
Socket	Deprecated: The connected CGI socket (only in: 2.3.4)	INFO
Request	Deprecated: The CGI request descriptor (only in: 2.3.4)	DEBUG

Name	Description	Since
MaconomyServer:WebComm	Issued when a WebComm command is received by the Maconomy Server (Java client, Analyzer or MScript). Requires the 'WebComm' TimeLog logger to be enabled.	2.3.5, 2.4.4
Key	Value	Log Level
Context	The WebComm command string	INFO

Name	Description	Since
MaconomyServer:MScript	Issued when an MScript command is received by the server. Requires the 'MScript' TimeLog logger to be enabled.	2.3.5, 2.4.4
Key	Value	Log Level
Context	The name of the MScript command	INFO

Container Requests

The following container request-related log entries exist:

Name	Description	Since
Container:Open	Issued when a container is opened.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO

Name	Description	Since
Container:Close	Issued when a container is closed.	2.3.4, 2.4.4
Issued when a container is closed.		
Key	Value	Log Level
ContainerName	The name of the container	INFO

Name	Description	Since
Container:Specify	Issued when a container specification is requested.	2.3.4, 2.4.4
Key	Value	Log Level

Log Events

ContainerName	The name of the container	INFO
---------------	---------------------------	------

Name	Description	Since
Container:Lock	Issued when a container entry is locked. (deprecated, only in 2.4)	2.3.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Unlock	Issued when a container entry is unlocked. (deprecated, only in 2.4)	2.3.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Initialize	Issued when a new container entry is initialized.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction (optional)	DEBUG
NewValues	'null'	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Create	Issued when a new container entry is created.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
NewValues	The new values to create the container entry with	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
------	-------------	-------

Log Events

Container:Read	Issued when a container entry is read.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Update	Issued when a container entry is updated.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
NewValues	The new values to update the container entry with	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Delete	Issued when a container entry is deleted.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Action	Issued when a container action is executed.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
ActionName	The name of the action	INFO
Restriction	The current container restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Print	Issued when a container print action is executed.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Restriction	The current container restriction	DEBUG

Log Events

Parameters	The current container parameters	DEBUG
------------	----------------------------------	-------

Name	Description	Since
Container:Move	Issued when a container table move operation is performed.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
Operation	The move operation being performed	INFO
SourceRestriction	The source row restriction	DEBUG
TargetRestriction	The target row restriction	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Restore	Issued when a container restore operation is performed.	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
RestoreData	The restore data (optional)	DEBUG
Parameters	The current container parameters	DEBUG

Name	Description	Since
Container:Restrict	Issued when a restriction is acquired for the current container entry	2.3.4, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
ForeignKeyName	The name of the foreign key that provides the restriction	DEBUG
RestrictionData	The record data to base the restriction on	DEBUG
Parameters	The current container parameters	DEBUG

Container Contributions

The following log entry breaks down container request-related time into its container contribution pre/post phases:

Name	Description	Since
Container:Contribution	<p>Issued when a container contribution event phase (pre/post) is executed.</p> <p>The execution of a container contribution event is split into 3 phases:</p> <ol style="list-style-type: none"> 1. A 'pre' event phase performed by the container contribution (PhaseId="PRE") 2. A delegation to the underlying container contributions 3. A 'post' event phase performed by the container contribution (PhaseId="POST") <p>The total time spent on an event in a container contribution is the sum of the 'pre' and 'post' phase durations logged for that container contribution. The time spent in underlying container contributions is not included.</p> <p>A container contribution is uniquely identified by its container name, contribution id and bundle id.</p>	2.3.5, 2.4.4
Key	Value	Log Level
ContainerName	The name of the container	INFO
EventId	<p>The ID of the container event being processes:</p> <p>OPEN, CLOSE, SPECIFY, TRANSACTION, INITIALIZE, CREATE, READ, UPDATE, DELETE, ACTION, PRINT, MOVE, RESTRICT</p>	INFO
ContributionId	The contribution ID of the container contribution	INFO
BundleId	The bundle ID of the container contribution	INFO
PhaseId	<p>The container contribution phase being executed:</p> <p>PRE, POST</p>	INFO
ActionName	The name of the action being executed, if EventId="ACTION" (optional)	INFO

Remote Procedure Calls

The following log entries are issued when RPC functions are invoked:

Name	Description	Since
MaconomyServer:Request	Issued when a call to an RPC function is performed. Required log level: DEBUG	2.3.4, 2.4.4
Key	Value	Log Level
PID	The PID of the remote server process	DEBUG
Function	The RPC function being called	DEBUG
Parameters	The parameters to the RPC function	TRACE

Name	Description	Since
MaconomyServer:Callback	Issued when an RPC callback function invocation is received from the remote server. Required log level: DEBUG	2.3.4, 2.4.4
Key	Value	Log Level
PID	The PID of the remote server process	DEBUG
Function	The RPC callback function being invoked	DEBUG
Parameters	The parameters to the RPC callback function	TRACE

Name	Description	Since
MaconomyServer:RPC	Issued when an RPC function call is executed on the remote server.	2.3.5, 2.4.4
Key	Value	Log Level
Context	The RPC function being called. The 'Enter' event will have the format: <i>'functionName(parameters)'</i> The 'Exit' event will have the format: <i>'functionName(parameters)->(returnValues)'</i>	INFO
PID	The PID of the remote server process	INFO

MSL Requests

The following log entry is produced by the 'MSL' TimeLog logger.

Name	Description	Since
------	-------------	-------

Log Events

MaconomyServer:MSL	Issued when an MSL dialog script or sub-program script is invoked	2.3.5, 2.4.4
Key	Value	Log Level
Context	The name of the MSL script	INFO

Database Requests

The following log entries are produced by the 'Database' TimeLog logger.

Name	Description	Since
MaconomyServer:Database	Issued when an database operation is invoked	2.3.4, 2.4.4
Key	Value	Log Level
Context	<i>Op: Query</i>	INFO
-where 'Op' can be:		
Cursor:Declare	Cursor declare operation	
Cursor:Execute	Cursor execute operation	
Cursor:Fetch[{Count}]	Cursor fetch operation. This log entry can appear in two variations: As a log entry for the execution time of a single database fetch operation As an accumulating log entry with the elapsed time since the first record was fetched from the cursor, and the total number of records fetched The accumulating log entry is only generated if the cursor is fetched until exhaustion (i.e. the last fetch operation returns 'nil')	
Cursor:Update	Cursor update operation	
Cursor:Insert	Cursor insert operation	
Cursor:Delete	Cursor delete operation	
SqlRequest	A "raw" (non-cursor) SQL request operation	

Security Log

The following log entries are produced by the 'SecurityLog' TimeLog logger.

Name	Description	Since
MaconomyServer:SecurityLog	Issued when a security log event is created. All login attempts, whether successful or not, will create a security log event.	2.3.4, 2.4.4
Key	Value	Log Level
Context	<i>Message [User]</i>	INFO
Audit	The Audit flag is always set for this event	n/a

Callbacks

The following callback-related log entries exist:

Name	Description	Since
RequestContext:Callback	Deprecated: Replaced by Callback:Function Issued when a callback is dispatched out of the server execution context.	2.3.4 only

Log Events

Value	Description	Log Level
Function	The callback function	INFO
Result	The resulting value returned from the callback handler (only on the 'Exit' log entry)	INFO

Name	Description	Since
Callback:Function	Issued when a callback is dispatched out of the server execution context. The time duration logged for a callback is time that the server process has been idle, waiting for a response from the client.	2.3.5, 2.4.4
Key	Value	Log Level
Function	The callback function	INFO
Result	The resulting value returned from the callback handler (optional)	INFO

Name	Description	Since
Callback:Progress	<p>Issued when a progress callback is dispatched out of the server execution context.</p> <p>The progress callback log entry covers the entire life cycle of the progress callback, i.e. from 'ShowProgress' across all 'UpdateProgress' calls to the final call to 'RemoveProgress'.</p> <p>This allows the log entry to exceed an entry duration threshold even though the individual progress callbacks are fast. Each 'UpdateProgress' call is logged as a 'Message'-type log entry and will contain the reported "work completed" fraction.</p> <p>The time spent in a progress callback is reported as 'Elapsed' rather than as 'Duration', since this time does not count towards the total execution time of the request.</p>	2.3.5, 2.4.4
Key	Value	Log Level
Message	The progress message	INFO
Fraction	The "work completed" fraction (only for 'Message'-type entries)	INFO
Result	The resulting value returned from the callback handler (only on the 'Message' and 'Exit' log entries, and for the 'Message' entries showing the return value of the previous progress callback)	INFO

Notifications

When actions are performed in the context of a notification recalculation, the threshold log level for log entries is raised to TRACE, to avoid filling the log output with highly repetitive universe queries.

This means that further log entries for notification recalculation-related operations will only be printed if the log level of the Logback logger is also set to TRACE.

Sample Queries

In this section we will show some examples of how the monitor log entries can be queried. The queries will be written in a SQL-like syntax, and should be fairly easy to translate to the concrete query syntax of any capable log reporting framework.

In the example the table of log entries is referred to as 'LogEntries'.

Request Execution Time

To list all requests and their execution time simply group the requests by 'RequestId' and add up the 'Duration' of the top-level log entries in each request:

```
SELECT RequestId, COALESCE(SUM(Duration),0) FROM LogEntries
WHERE Type = 'Exit' AND LogDepth = 0
GROUP BY RequestId;
```

Notice the use of the 'COALESCE(...)' function to normalize NULL values. In this example it's a highly theoretical possibility, but as we begin to introduce various sub-queries the consistent use of this function becomes very important.

The execution times returned by the query above will include time spent waiting for user callbacks, which is typically not relevant from a performance perspective. To take this into account the query can be expanded to subtract the time spent in callbacks:

```
SELECT RequestId, TotalTime, CallbackTime, TotalTime-CallbackTime AS RequestTime FROM
(SELECT
  RequestId,
  TotalTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
    WHERE Type = 'Exit' AND LogDepth = 0 AND RequestId = L.RequestId),
  CallbackTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
    WHERE Type = 'Exit' AND NAME LIKE 'Callback:%' AND RequestId = L.RequestId)
FROM LogEntries L
GROUP BY RequestId) Result;
```

The query can be expanded further to also report the time spent on database operations:

```
SELECT RequestId, TotalTime, CallbackTime, DatabaseTime, TotalTime-CallbackTime AS
RequestTime FROM
(SELECT
  RequestId,
  TotalTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
    WHERE Type = 'Exit' AND LogDepth = 0 AND RequestId = L.RequestId),
  CallbackTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
    WHERE Type = 'Exit' AND NAME LIKE 'Callback:%' AND RequestId = L.RequestId),
  DatabaseTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
    WHERE Type = 'Exit' AND NAME LIKE '%:Database' AND RequestId = L.RequestId)
FROM LogEntries L
```

```
GROUP BY RequestId) Result;
```

Workspace Requests

The following query example shows how to extract information on all Workspace Client requests:

```
SELECT
    RequestId, WorkspaceName, PaneRequest, Duration
FROM LogEntries
WHERE Type = 'Exit' AND Name LIKE 'Workspace:%';
```

The only caveat in this query is that it only shows the time spent on the workspace-specific part of the request, which is not necessarily representative for the total execution time of the request. Luckily the query can easily be amended to also calculate the total execution time:

```
SELECT
    RequestId, WorkspaceName, PaneRequest, Duration,
    TotalTime = (SELECT COALESCE(SUM(Duration),0) FROM LogEntries
        WHERE RequestId = L.RequestId AND LogDepth = 0)
FROM LogEntries L
WHERE Type = 'Exit' AND Name LIKE 'Workspace:%';
```

WebService and WebDaemon Requests

The following query examples show how to extract information on WebService and WebDaemon requests. These queries follow the same pattern as for Workspace Requests discussed above, except that the additional inner select used to obtain the total execution time is not necessary, since these request types at the time of writing are always logged at top-level (LogDepth = 0):

```
SELECT
    RequestId, Method, Url, Query, Duration
FROM LogEntries
WHERE Type = 'Exit' AND Name LIKE 'WebService:%';

SELECT
    RequestId, Method, Url, Query, Request, RemoteAddr, Duration
FROM LogEntries
WHERE Type = 'Exit' AND Name LIKE 'WebDaemon:%';
```

Login Auditing

The following query examples show how to extract information about user logins.

If the 'SecurityLog' TimeLog logger is enabled an audit-level log event is issued for every login attempt:

```
SELECT
    Context, PID
FROM LogEntries
WHERE Type = 'Exit' AND Name LIKE ':%SecurityLog';
```

In addition, login and logout requests from all Maconomy 2.x clients (Workspace Client, REST-API) will trigger 'Auth' events in the Coupling Service:

```
SELECT
    LoginRules, PrincipalName
FROM LogEntries
WHERE Type = 'Exit' AND Name = 'Auth:Login' OR Name = 'Auth:Logout';
```

Active Request

To identify requests which have not yet completed, we must find requests that have a top-level 'Enter' entry without a matching 'Exit' entry. This can be as simple as selecting all 'Enter' entries with log depth zero and subtract all 'Exit' entries with log depth zero:

```
SELECT RequestId FROM LogEntries WHERE Type = 'Enter' AND LogDepth = 0
EXCEPT
SELECT RequestId FROM LogEntries WHERE Type = 'Exit' AND LogDepth = 0
```



About Deltek

Better software means better projects. Deltek is the leading global provider of enterprise software and information solutions for project-based businesses. More than 23,000 organizations and millions of users in over 80 countries around the world rely on Deltek for superior levels of project intelligence, management and collaboration. Our industry-focused expertise powers project success by helping firms achieve performance that maximizes productivity and revenue. www.deltek.com