Java agent configuration: Config file

On this page

☰On this page

The New Relic Java agent reads its configuration from the `newrelic.yml` file. By default the agent looks for this file in the directory that contains `newrelic.jar`. You can override the config file's location by setting the `newrelic.config.file` system property to a fully qualified file name.

Configuration file structure

The `newrelic.yml` file is split into stanzas corresponding to different environments:

- Test
- Development

- Staging
- Production (default)

New Relic applies settings in the common stanza to each of these environments. You can select other environments as the default by setting the `newrelic.environment` system property to the environment name.

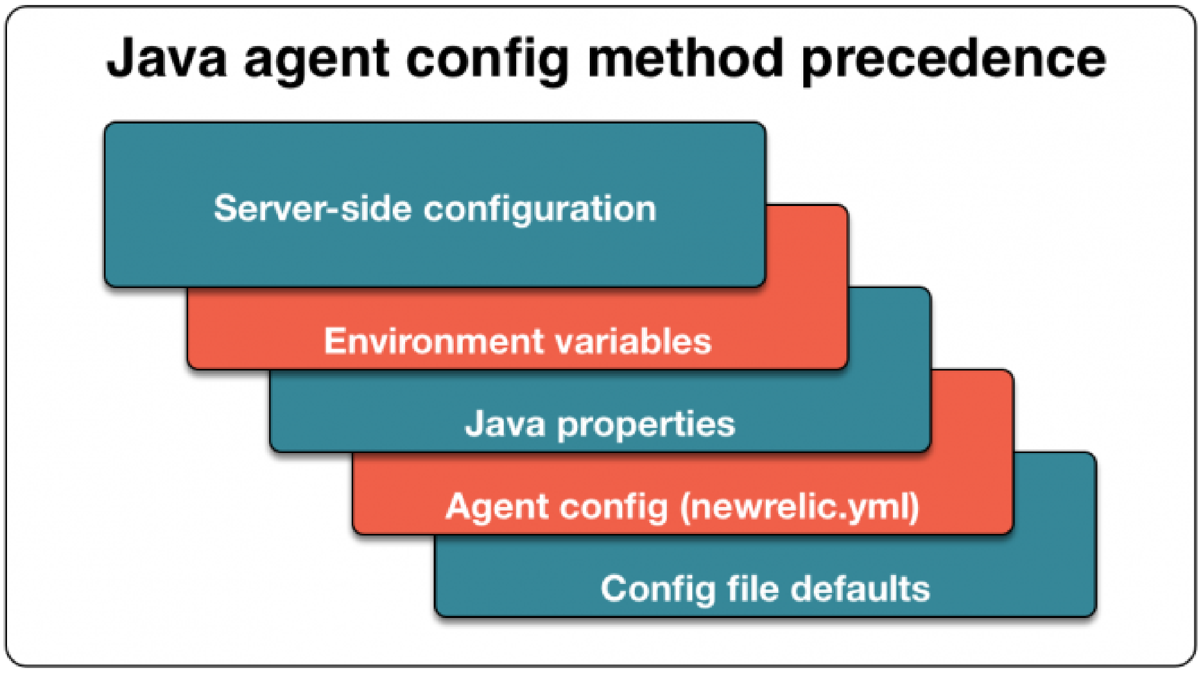
A `newrelic.yml` template is available.

If you edit `newrelic.yml`, be careful to conform to the YAML format. Use a YAML validator to ensure the syntax is accurate before using the file with New Relic's Java agent, and follow these rules:

Java agent newrelic.yml	Requirements
Format	YML files are case sensitive.
Indentations	All indentations must be in increments of two characters. Other indentations will result in an `Unable to parse configuration file` error upon agent startup. • Use the same level of indentation for data in the same stanza of the file. • Indent any sub-stanzas by an additional two spaces.
Changes to file	You must restart your JVM host process for changes to take effect. **Exception:** Property changes to `log_level` and `audit_mode` do not require a restart. Property changes under circuit breaker don't require a restart.

Configuration settings precedence

To override any setting in the config file, use a system property override. In certain environments, environment variables can also be used to override both the config file and the system properties. The environment variables primarily exist to support Heroku. When used, server-side configuration overrides all other configuration settings.


Java agent config method precedence
Server-side configuration
Environment variables
Java properties
Agent config (newrelic.yml)
Config file defaults

With the Java agent, server-side configuration overrides all other settings. Environment variables override Java system properties. Java properties override user configuration settings in your `newrelic.yml` file. User settings override the `newrelic.yml` default settings.

Configuring the Java extensions directory

The Java agent reads the configuration files on process startup. To identify the directory where the files are located, either create a new or specify an existing extensions directory:

Hide All

Create an extensions directory

To create the extensions directory:

1. Navigate to the directory where `newrelic.jar` and `newrelic.yml` are located. Create a directory named `extensions`.
2. In `newrelic.yml`, check that the property `extensions.dir` is not set.

Specify an existing extensions directory

To use an existing Java extensions directory:

1. In your `newrelic.yml` file, locate the `common` section.
2. Use the property `extensions.dir` to specify the location of the file.

General configuration settings

Set these options in the `common` stanza. To override any of these options, use a `newrelic.config` prefixed system property.

Hide All

license_key (REQUIRED)

Type	String
Default	(none)

This setting is **required**. You must specify the license key associated with your New Relic account. This key binds your agent's data to your account in the New Relic service.

app_name (REQUIRED)

Type	String
Default	(none)

This setting is **required**. Defines the application name used to report data to New Relic.

If `enable_auto_app_naming` is false, the agent reports all data to this application. Otherwise, the agent reports only background tasks (transactions for non-web applications) to this application.

To report data to more than one application, separate the application names with a semicolon. For example, to report data to **My Application** and **My Application 2** use this:

```
app_name: My Application;My Application 2
```

For more methods of naming your application, see Name your Java application.

agent_enabled

Type	Boolean
Default	`true`

Flag to enable the agent. Use this setting to force the agent to run or not run.

apdex_t (DEPRECATED)

Type	Float
Default	`1.0`

The `apdex_t` threshold in seconds for the application's Apdex score. For Java agent versions 1.2.008 or higher, the `apdex_t` value is set in the UI and the value in `newrelic.yml` is ignored.

appserver_port

Type	Integer
Default	(none)

Number to differentiate JVMs for the same app on the same machine. New Relic uses host/port for uniqueness, so you can distinguish the JVMs by putting a switch like this into the startup arguments for each JVM:

```
-Dnewrelic.config.appserver_port=8081
```

Once you have used `appserver_port` to name the JVMs and restart them, you should be able to see them individually in the dropdown and in the profiling interface.

This is only a change for New Relic; it doesn't actually affect the port on which the host communicates in any way.

audit_mode

Type	Boolean
Default	`false`

Enables plain text logging of all data sent to New Relic to the agent logfile. This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

ca_bundle_path

Type	String
Value Format	*`/path/to/ca/cert/bundle.pem`*

Specifies a path to a custom SSL certificate bundle that will be used by the agent to establish a secure connection to New Relic. If your custom SSL certificate bundle doesn't include certificates that are sufficient to connect to New Relic then you'll need to merge the required certs into your custom certificate bundle.

use_private_ssl

Type	Boolean
Default	`false`

The following SSL certificates are bundled into the agent jar:

```
META-INF/certs/eu-newrelic-com.pem
META-INF/certs/eu01-nr-data-net.pem
META-INF/certs/newrelic-com.pem
```

By default (`use_private_ssl: false`) the agent will use the SSL certificates bundled into the JDK to establish a secure connection to New Relic or the custom SSL certificates bundle specified by `ca_bundle_path` . If you want to use the SSL certificates bundled with the agent, set `use_private_ssl: true` .

Note: `use_private_ssl` will be ignored if `ca_bundle_path` is set.

enable_auto_app_naming

Type	Boolean
Default	`false`

Enables the reporting of data separately for each web app. Set to `true` to enable support for auto app naming. The name of each web app is detected automatically and the agent reports data separately for each one. This provides a finer-grained performance breakdown for web apps in New Relic.

For more information, see Automatic application naming.

For more methods of naming your application, see Name your Java application.

enable_auto_transaction_naming

Type	Boolean
Default	`true`

Enables component-based transaction naming. Set to `true` to enable component-based transaction naming. Set to `false` to use the URI of a web request as the name of the transaction. For more information, see Naming web transactions.

Unless you implement API calls to name your transactions, disabling auto-transaction naming is very likely to cause Metric grouping issues.

enable_custom_tracing

Type	Boolean
Default	`true`

Enables all instrumentation using an `@Trace` annotation. Disabling this causes `@Trace` annotations to be ignored.

extensions.dir

Type	String
Default	(none)

Defines the location of the optional extensions directory. If this property is not set, the agent will look for a subdirectory named `extensions` in the same directory as `newrelic.jar` and `newrelic.yml` .

high_security

Type	Boolean
Default	`false`

In order for high security to be enabled, this property must be set to `true` and the high security property in the New Relic user interface must be enabled. Enabling high security means SSL is turned on, request and message queue parameters are not collected, and queries cannot be sent to New Relic in their raw form.

As of Java agent 3.48.0, SSL is enabled by default and the config option to disable it has been deprecated. As of Java agent 4.0.0, the ability to disable SSL has been removed.

insert_api_key

Type	String
Default	(none)

A valid Insert API Key for your account. This is only required for Real-time Java profiling using JFR metrics.

labels

Type	String
Default	`""`

Attach tags to this app.

Note that this option now enables tags, which replaced the label feature. You can still query your historical labels.

max_stack_trace_lines

Type	Integer
Default	`30`

Limits the number of lines the agent collects from each stack trace. Increasing this value may impact performance, because it increases the amount of memory the agent uses and the amount of data sent to New Relic.

proxy_host

Type	String

Default	(none)

The proxy host through which to connect to the New Relic collector. If a proxy is used, the host setting is required. Other proxy settings are optional.

proxy_password

Type	String
Default	(none)

The password for proxy authentication. If a proxy is used, the host setting is required. Other proxy settings are optional. The username and password settings will be used to authenticate to Basic Auth challenges from a proxy server.

> The Java agent supports Basic (clear text) authentication.

proxy_port

Type	String
Default	`8080`

The proxy host port number. If a proxy is used, the host setting is required. Other proxy settings are optional.

proxy_user

Type	String
Default	(none)

The username for proxy authentication, such as Basic (clear text) authentication. If a proxy is used, the host setting is required. Other proxy settings are optional. The username and password settings will be used to authenticate to Basic Auth challenges from a proxy server.

proxy_scheme

Type	String
Default	(none)

The proxy scheme used. Setting `proxy_scheme: "https"` will allow the agent to connect through proxies using the HTTPS scheme.

reactor-netty.errors.enabled

Type	Boolean
Default	`true`

Whether errors are reported for reactor netty. If set to false, errors will be ignored.

Only available in Java agent 6.3.0 and above.

send_data_on_exit

Type	Boolean
Default	`false`

Enable delayed JVM shutdown to give the agent a chance to send latest metric data to New Relic before JVM shutdown.

send_data_on_exit_threshold

Type	Integer
Default	`60`

The number of seconds after which the agent will use the `send_data_on_exit` setting.

send_environment_info

Type	Boolean
Default	`true`

Enable reporting of JVM settings to New Relic.

send_jvm_props

Type	Boolean
Default	`true`

When set to `true`, JVM properties will be sent to New Relic.

ssl (DEPRECATED)

Type	Boolean
Default	`true`

Requires connections to the New Relic collector to go over SSL.

The agent communicates with New Relic via HTTPS by default, and New Relic requires HTTPS for all traffic to APM and the New Relic REST API.

This work is done asynchronously to the threads that process your application code, so response times will not be directly affected by this change.

As of Java agent 3.48.0, SSL is enabled by default and the config option to disable it has been deprecated. As of Java agent 4.0.0, the ability to disable SSL has been removed.

sync_startup

Type	Boolean
Default	`false`

Enable the agent to connect the New Relic collector immediately upon app startup.

scala_futures_as_segments

This applies to Java agent version 3.44.0 or higher.

Type	Boolean
Default	`false`

For more concise transaction trace details, the agent does not report Scala Futures as transaction segments, and those Futures do not contribute to the transaction's total time.

Enabling `scala_futures_as_segments` increases your overhead. If you want Scala Futures to report as transaction segments so you can view them in a transaction trace, you can enable it:

```
scala_futures_as_segments:
  enabled: true
```

Logging configuration

These are part of the general configuration variables. They are broken out here because they are frequently tweaked for debugging.

Some of the logging configuration variables are dynamic and do not need a host restart for them to take effect. For instance, if log files are growing too quickly, `log_level` can be set to a less verbose setting to reduce the reporting rate.

Here is the order of precedence for configuration variables affecting log rotation.

- If `log_daily` is `true`, other log rotation settings are ignored.
- If `log_file_count` is `1` or `0`, the size limit is ignored.
- Finally, the agent applies `log_limit_in_kbytes`.

Depending on the growth rate, it is possible for the log file size to exceed the configured value by a small amount.

Hide All

log_daily

Type	Boolean
Default	`false`

Set to `true` to roll the logs daily. Overrides the other configuration variables that affect log rotation.

log_file_count

Type	Integer
Default	`1`

The maximum number of log files to keep when using log rotation.

log_file_name

Type	String
Default	`newrelic_agent.log`

The unqualified log file name or the string `STDOUT` which will log to standard out.

log_file_path

Type	String
Default	`logs` subdirectory where `newrelic.jar` is located

The log file path.

If `log_file_path` is specified, the directory must already exist. If the default value is used, the agent will attempt to create the directory.

log_level

Type	String
Default	`info`

The log verbosity level.

The agent uses its own log file to keep its logging separate from that of your application. Valid options, in order of verboseness, are:

- `off`
- `severe`
- `warning`
- `info`
- `fine`
- `finer`
- `finest`

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

log_limit_in_kbytes

Type	Integer
Default	`0`

The log file size in kilobytes at which log files are rotated. Set to `0` for no limit.

JMX

To set these options, use the `jmx` stanza. To override them, use a `newrelic.config.jmx` prefixed system property.

The Java agent uses JMX to collect JVM data as well as to communicate with the JFR (Java Flight Recorder) daemon for Real-time Java profiling.

Hide All

enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all JMX functionality.

linkingMetadataMBean

Type	Boolean
Default	`false`

This setting can be enabled to allow the Java agent to expose linking metadata to the JFR daemon. Doing so allows the JFR daemon to obtain the entity GUID generated by the

Java agent and link JFR data with the same APM application that is being monitored by the agent instead of as a separate entity.

> This applies to Java agent version 6.1.0 or higher.

Attributes

To set these options, use the `attributes` stanza. To override them, use a `newrelic.config.attributes` prefixed system property.

Attributes are key-value pairs that provide information for transaction traces, traced errors, browser monitoring, and transaction events. There is also an attribute stanza under each destination. For more information, see Java agent attributes, Enabling and disabling attributes and Attribute examples.

Hide All

enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes.

> For security reasons, capturing custom attributes using the Custom Instrumentation Editor is set to `false` by default.

include

Type	List of Strings
Default	(none)

If attributes are enabled, attribute keys found in this list will be sent to New Relic. Separate the keys in the list with a comma; for example:

```
key1, key2, key3
```

Also refer to the agent attribute rules.

exclude

Type	List of Strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic. Separate the keys in the list with a comma; for example:

```
key1, key2, key3
```

Also refer to the agent attribute rules.

Transaction tracer

These options are set in the `transaction_tracer` stanza and can be overridden by using a `newrelic.config.transaction_tracer` prefixed system property.

Transaction tracing captures deep information about slow transactions and sends this to the New Relic service. The transaction includes the exact call sequence of the transactions, including any query statements issued.

> Do not use brackets `[suffix]` at the end of your transaction name. New Relic automatically strips brackets from the name. Instead, use parentheses `(suffix)` or other symbols if needed.

Hide All

enabled

Type	Boolean
Default	`true`

The transaction tracer is enabled by default. Set this to `false` to turn it off.

explain_enabled

Type	Boolean
Default	`true`

Determines whether the agent will capture the `EXPLAIN` plan for slow queries. Only supported for MySQL and PostgreSQL.

explain_threshold

Type	Float
Default	`0.5`

Threshold in seconds for query execution time below which the slow query and the `EXPLAIN` plan (if supported) will not be captured. Relevant to slow queries only when `record_sql` is set to `raw` or `obfuscated`. Relevant to `EXPLAIN` plans only when `explain_enabled` is set to `true`.

insert_sql_max_length

Type	Integer
Default	2000

The character limit for the SQL query string. If you have many slow SQL queries with large chunks of information, this could negatively affect performance or how quickly you see your data in New Relic. Increase the value gradually until you find the right balance of information and performance.

log_sql

Type	Boolean
Default	`false`

Set to `true` to enable logging of queries to the agent log file instead of uploading to New Relic. Queries are logged using the `record_sql` mode.

record_sql

Type	String
Default	`obfuscated`

When the transaction tracer is on, query statements can optionally be recorded. The recorder has three modes:

- `off`: Send no queries.
- `raw`: Send the query statement in its original form.
- `obfuscated`: Strips out numeric and string literals.

stack_based_naming (Play 2.x+ only)

Type	Boolean
Default	`False` Defaulted to `true` until Java agent version 3.12.1, when it was changed to `false`.

This option is for Play 2.x+ only. Play/Scala instrumentation can use `Thread.getStackTrace()` to improve tracer naming, but at the cost of increased overhead.

stack_trace_threshold

Type	Integer
Default	`0.5`

This is the threshold at which segments will be given a stack trace in the transaction trace.

top_n

Type	Integer
Default	`20`

Use this setting to control the variety of your transaction traces. `top_n` is an integer that represents the number of unique, slow transactions that traces will be created for.

- If you want transaction traces to more accurately reflect the actual slowest transactions in your app, make this value **lower**.
- If you want to sample a more diverse array of transactions, make the value **higher**.

A value of 0 would mean that **only** the slowest transaction is always traced. This is considered not to be optimal, though, because you may have one or two transactions that are always the slowest, and repeatedly seeing those same transaction traces will probably not give you much value.

If the same transaction is often the slowest, the `top_n` setting allows the Java agent (over time) to sample the slowest `n` transactions. This gives you greater variety and more insight into your application.

transaction_threshold

Type	String (float)
Default	`apdex_f`

The time threshold used to determine when a transaction is eligible to be traced. When the transaction's response time exceeds this threshold, a transaction trace will be recorded and sent to New Relic.

The default is `apdex_f` (default), which sets the threshold to be the "Frustrated" Apdex level (four times the apdex_t value). You can also set a specific time threshold by entering a float value that represents a number of seconds.

slow_query_whitelist (DEPRECATED)

Type	String
Default	(none)

This config has been deprecated as of agent version 5.10.0 and will be removed in a future agent version. Instead use `collect_slow_queries_from`.

By default, high security mode does not allow the agent to collect slow queries. Enable this option to collect Cassandra queries from the DataStax driver, even with high security enabled. If you don't use high security, the agent collects slow queries automatically.

For DataStax driver 2.1.2, add this rule to your allow list:

```
transaction_tracer:
    slow_query_whitelist:
     'com.newrelic.instrumentation.cassandra-datastax-2.1.2'
```

For DataStax driver 3.0.0, add this rule to your allow list:

```
transaction_tracer:
    slow_query_whitelist:
     'com.newrelic.instrumentation.cassandra-datastax-3.0.0'
```

collect_slow_queries_from

Type	String
Default	(none)

By default, high security mode does not allow the agent to collect slow queries. Enable this option to collect Cassandra queries from the DataStax driver, even with high security enabled. If you don't use high security, the agent collects slow queries automatically.

For DataStax driver 2.1.2, add this rule to your allow list:

```
transaction_tracer:
    collect_slow_queries_from:
     'com.newrelic.instrumentation.cassandra-datastax-2.1.2'
```

For DataStax driver 3.0.0, add this rule to your allow list:

```
transaction_tracer:
    collect_slow_queries_from:
     'com.newrelic.instrumentation.cassandra-datastax-3.0.0'
```

attributes.enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes for transaction traces. If `attributes.enabled` at the root level is `false`, no attributes will be sent to transaction traces regardless on how this property (`transaction_tracer.attributes.enabled`) is set.

attributes.include

Type	List of strings
Default	(none)

If attributes are enabled for transaction traces, all attribute keys found in this list will be sent to New Relic in transaction traces. For more information, see the agent attribute rules.

attributes.exclude

Type	List of Strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic in transaction traces. For more information, see the agent attribute rules.

token_limit

Type	Integer
Default	3000

Limits the number of Tokens that can be created per Transaction. Increasing this value may impact performance, because it increases the amount of memory the agent uses and the amount of data sent to New Relic.

segment_limit

Type	Integer
Default	3000

Limits the number of segments that can be created per transaction. Increasing this value may impact performance, because it increases the amount of memory the agent uses and the amount of data sent to New Relic.

gc_time_enabled

Type	Boolean
Default	`true`

Records time spent waiting for garbage collection during the duration of a transaction and includes the GC time in the corresponding transaction trace. Enabled by default as of Java agent 5.2.0.

Transaction segments

These options are set in the `transaction_segments` stanza and can be overridden by using a `newrelic.config.transaction_segments` prefixed system property.

Transaction segments represent discrete pieces of work (generally method calls) and are displayed within transaction traces.

Transaction segment attribute filtering requires Java agent version 4.10.0 or higher.

Hide All

attributes.enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes for transaction segments. If `attributes.enabled` at the root level is `false`, no attributes will be sent to transaction segments regardless on how this property (`transaction_segments.attributes.enabled`) is set.

attributes.include

Type	List of strings
Default	(none)

If attributes are enabled for transaction segments, all attribute keys found in this list will be sent to New Relic in transaction segments. For more information, see the agent attribute rules.

attributes.exclude

Type	List of Strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic in transaction segments. For more information, see the agent attribute rules.

Browser monitoring

These options are set in the `browser_monitoring` stanza and can be overridden by using a `newrelic.config.browser_monitoring` prefixed system property.

Browser monitoring gives you insight into the performance real users are experiencing with your website. This is accomplished by measuring the time it takes for your users' browsers to download and render your web pages by injecting a small amount of JavaScript code into the header and footer of each page.

Hide All

auto_instrument

Type	Boolean
Default	`true`

By default the agent automatically inserts API calls in compiled JSPs to inject the monitoring JavaScript into web pages. Set this attribute to `false` to turn off this behavior.

disabled_auto_pages

Type	Comma-separated list of strings
Default	(none)

When `auto_instrument` is `true`, by default all pages are instrumented. List all pages that you want the auto instrumentation to skip here. You can still use manual instrumentation on these pages.

For example:

```
    browser_monitoring:
      disabled_auto_pages: /WEB-INF/jsp/testpage_1.jsp, /WEB-INF/jsp/testpage_2.jsp
```

attributes.enabled

Type	Boolean
Default	`false`

This setting can be used to turn on or off all attributes for browser monitoring. This is the data you can query. If `attributes.enabled` is false at the root level, no attributes will be sent up in browser monitoring regardless on how this property under `browser_monitoring` is set.

attributes.include

Type	List of Strings
Default	(none)

If attributes are enabled for `browser_monitoring`, all attribute keys found in this list will be sent to New Relic in page views. For more information, see the agent attribute rules.

attributes.exclude

Type	List of Strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic in page views. For more information, see the agent attribute rules.

External tracer

The external tracing options are set in the `external_tracer` stanza and can be overridden by using a `newrelic.config.external_tracer` prefixed system property.

Hide All

exclude_request_uri

Type	Boolean
Default	`false`

This setting can be used to control the collection of outgoing request URIs for errors and transaction traces. Set this to true to disable collecting this information.

Cross application tracer

The cross application tracing options are set in the `cross_application_tracer` stanza and can be overridden by using a `newrelic.config.cross_application_tracer` prefixed system property.

Cross application tracing adds request and response headers to external calls using the Apache HttpClient libraries. This provides better performance data when calling applications monitored by other New Relic Agents.

Hide All

enabled

Type	Boolean
Default	`true`

Cross application tracing is enabled by default. Set this to false to turn it off.

Error collector

These options are set in the `error_collector` stanza and unless noted otherwise can be overridden by using a `newrelic.config.error_collector` prefixed system property. The error collector captures information about uncaught exceptions and sends them to New Relic for viewing.

For how to configure errors for the Java agent, including how to configure errors via the UI, see Java agent error configuration.

Hide All

enabled

Type	Boolean
Default	`true`

Enable error collection.

ignore_classes

Type	Stanza containing a list of fully qualified `class_name` strings
Default	(none)

Specified exception class names will be ignored and will not affect error rate or Apdex score, or be reported to APM. **Cannot be specified by system property.**

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
    error_collector:
      ignore_classes:
        - "com.example.MyException"
        - "com.example.DifferentException"
```

ignore_messages

Type	Stanza containing a fully qualified `class_name` and a list of `messages` per error class
Default	(none)

Specify exception class names the agent should ignore. Ignored messages will not affect error rate or Apdex score, or be reported to APM. Contains `yaml` pairs consisting of:

- A fully qualified exception class name that should not be reported to APM

  AND
- A list of exception `message` s to match against (at least one is required)

If the exception class name matches an error but the message does not, then that error **will not** be ignored. Message strings use `contains` for matching. A message cannot be provided on its own and must always be paired with a fully qualified class name. **Cannot be specified by system property.**

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
error_collector:
  ignore_messages:
    com.example.MyException:
      - "Some error message to ignore"
      - "Some other error message to ignore"
    com.example.DifferentException:
      - "Some different error message to ignore"
```

ignore_status_codes

Type	Comma-separated list of strings and ranges
Default	`404`

A comma-separated list comprised of individual and dashed ranges of HTTP status codes that should not be treated as errors.

If this property is commented out in the `newrelic.yml` configuration file, then the 404 status code will automatically be ignored. When using server-side configuration, the status code 404 must be specified in order for it to be ignored.

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
error_collector:
  ignore_status_codes: 404,507-511
```

expected_classes

Type	Stanza containing a List of fully qualified `class_name` strings
Default	(none)

Prevents specified exception classes from affecting error rate or Apdex score while still reporting the errors to APM. **Cannot be specified by system property.**

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
error_collector:
  expected_classes:
    - "com.example.MyException"
    - "com.example.DifferentException"
```

expected_messages

Type	Stanza containing a fully qualified `class_name` and a List of `messages` per error class
Default	(none)

Contains yaml pairs consisting of a fully qualified exception class name that should be marked as expected and thus prevented from affecting error rate or Apdex score and a List of exception `message`s to match against, the latter of which at least one is required. If the exception class name matches an error but the message does not, then that error **will not** be marked as expected and therefore will affect error rate and Apdex score.

Message strings use `contains` for matching. A message cannot be provided on its own and must always be paired with a fully qualified class name. **Cannot be specified by system property.**

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
error_collector:
  expected_messages:
    com.example.MyException:
      - "Some expected error message"
      - "Some other expected error message"
    com.example.DifferentException:
      - "Some different expected error message"
```

expected_status_codes

Type	Comma-separated list of strings and ranges
Default	(none)

A comma-separated list comprised of individual and dashed ranges of HTTP status codes to be marked as expected and thus prevented from affecting error rate or Apdex score.

This setting is dynamic, so running agents will notice changes to `newrelic.yml` without a JVM restart.

For example:

```
error_collector:
  expected_status_codes: 415,500-506
```

attributes.enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes for traced errors. If `attributes.enabled` is `false` at the root level, then no attributes will be sent to traced errors regardless of how this property is set under `error_collector`.

attributes.include

Type	List of strings
Default	(none)

If attributes are enabled for traced errors, all attribute keys found in this list will be sent to New Relic in traced errors. For more information, see the agent attribute rules.

attributes.exclude

Type	List of strings
Default	(none)

Attribute keys found in this list will not be sent to New Relic in traced errors. For more information, see the agent attribute rules.

ignore_errors (DEPRECATED)

Type	Comma-separated list of Strings
Default	(none)

All specified exception class names specified will not be treated as errors. Deprecated as of Java agent 3.40.0 and replaced by `ignore_classes`.

For example:

```
error_collector:
  ignore_errors: some.other.MyException
```

Strip exceptions

These options are set in the `strip_exception_messages` stanza and unless noted otherwise can be overridden by using a `newrelic.config.strip_exception_messages` prefixed system property. This configuration can be enabled to control whether Java exception messages are reported to New Relic.

Hide All

enabled

Type	Boolean
Default	`false`

By default, this is set to `false`, which means that the agent sends messages from all exceptions to the New Relic collector.

- If you set this to `true`, the agent strips the messages from exceptions in order to prevent it from inadvertently capturing sensitive information.
- If you enable high security mode, this is automatically set to `true`.
- If you set `enabled` to `true` but you want the agent to capture messages from specific exceptions, add the exceptions to your allow list.

whitelist (DEPRECATED)

Type	String
Default	(none)

This config has been deprecated as of agent version 5.10.0 and will be removed in a future agent version. Instead use `allowed_classes`.

If you set `enabled` to `true` but you want the agent to capture messages for specific exceptions, add each exception to the `whitelist`, separated by a comma.

allowed_classes

Type	String
Default	(none)

If you set `enabled` to `true` but you want the agent to capture messages for specific exceptions, add each exception to `allowed_classes`, separated by a comma.

Thread profiler

These options are set in the `thread_profiler` stanza and can be overridden by using a `newrelic.config.thread_profiler` prefixed system property.

Thread profiler measures wall clock time, CPU time, and method call counts in your application's threads as they run.

Hide All

enabled

Type	Boolean
Default	`true`

Enable the thread profiler.

Transaction events

These options are set in the `transaction_events` stanza and can be overridden by using a `newrelic.config.transaction_events` prefixed system property.

Transaction events provide the data for displaying histograms and percentiles in the UI.

Previously this stanza was called `analytics_events`. If your configuration file still uses `analytics_events`, update your agent to use `transaction_events`.

Hide All

enabled

Type	Boolean
Default	`true`

Enable the transaction events service.

max_samples_stored

Type	Integer
Default	`2000`
Max	`10000`

The maximum number of sampled transaction events reported every 60 seconds.

custom_request_headers

Type	List of maps
Default	None

Unlike other settings, `custom_request_headers` have to be paired together and must be set in the `newrelic.yml` file. They can't be overwritten by Java virtual machine arguments (system property) or environment variables.

A list of maps with the paired keys `header_name` and the optional `header_alias. Choose one or more custom HTTP request headers to add as transaction attributes.`

You can list multiple header configurations:

```
    transaction_events:
       custom_request_headers:
           -
             header_name: "X-Custom-Header-1"
           -
             header_name: "X-Custom-Header-2"
             header_alias: "CustomHeader2alias"
```

In the first map set, `X-Custom-Header-1` is captured and reported by the agent as the header name for a corresponding value from the request object. The `header_name` will also be the name of the attribute sent to New Relic.

In the second map set, the request header is `X-Custom-Header-2`, but the `CustomHeader2alias` is the name sent to New Relic.

attributes.enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes for transaction events. If `attributes.enabled` is `false` at the root level, then no attributes will be sent to transaction events regardless of how this property is set under `transaction_events`.

attributes.include

Type	List of Strings
Default	(none)

If attributes are enabled for transaction events, all attribute keys found in this list will be sent to New Relic in transaction events. For more information, see the agent attribute rules.

attributes.exclude

Type	List of Strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic in transaction events. For more information, see the agent attribute rules.

Custom events

Custom events are set in the `custom_insights_events` stanza and can be overridden by using a `newrelic.config.custom_insights_events` prefixed system property.

APM lets you record custom event data via the New Relic language agent APIs, which you can then query.

For Java agent versions prior to 4.1.0, the following YAML configuration is recognized:

```
custom_insights_events.enabled: true
custom_insights_events.max_samples_stored: 5000
```

For agent versions 4.1.0 and above, the YAML configuration uses the nested stanza formatting:

```
custom_insights_events:
  enabled: false
  max_samples_stored: 5000
```

Hide All

enabled

Type	Boolean
Default	`true`

This enables the custom event service.

max_samples_stored

Type	Integer
Default/Max	`10000`

The maximum number of sampled custom events reported every 60 seconds.

Hostname configuration

These options are set in the `process_host` stanza and can be overridden by using a `newrelic.config.process_host` prefixed system property.

These properties are used for configuring the hostname displayed in the UI:

Hide All

display_name

Type	String
Default	(none)

Set a display name to decorate the "host:port" label in the New Relic UI.

ipv_preference

Type	String
Default	`4`

If the hostname cannot be determined, then the IP address of the host will be used. This property determines whether the IPv4 or IPv6 address should be used. The default is IPv4.

Custom instrumentation

These options set in the `class_transformer` stanza and can be overridden by using a `newrelic.config.class_transformer` prefixed system property.

Hide All

trace_annotation_class_name

Type	String
Default	(none)

String containing the full class name of the annotation class the agent uses to determine which user-specified methods to instrument. For more information about custom annotations, see Java custom metric collection.

com.newrelic.instrumentation.servlet-user

Type	Boolean
Default	`false`

Enable this option to capture the `userPrincipal` name. This name is included as a transaction trace attribute, and can be queried.

System properties

You can override any setting in the `newrelic.yml` file by setting a system property. The system property corresponding to a given setting in the config file is the setting name prefixed by `newrelic.config`. For example, the system property for the `log_level` setting is `newrelic.config.log_level`.

For settings nested in stanzas, prepend the stanza name to the setting name. For example, the system property for the `enabled` setting in the transaction_tracer stanza is `newrelic.config.transaction_tracer.enabled`.

In addition to overriding configuration settings, the following system properties are recognized by the agent:

Hide All

newrelic.config.process_host.display_name

Type	String
Default	`(none)`

Set a display name to decorate the "host:port" label in the New Relic UI. Requires Java agent 3.17 or higher.

newrelic.config.file

Type	String
Default	(none)

String containing a fully qualified path to the newrelic configuration file. If empty, the agent assumes `newrelic.yml` is in the same directory as `newrelic.jar`.

newrelic.debug

Type	Boolean
Default	(none)

Enable debug logging.

newrelic.environment

Type	String
Default	(none)

String containing the environment configuration for the agent to use.

newrelic.home

Type	String
Default	(none)

String containing the home directory of agent. This defaults to the same directory as the agent jarfile.

newrelic.logfile

Type	String

Default	`newrelic_agent.log`

String containing the name of the agent log file.

Environment variables

Environment variables take the highest precedence and override the system properties and yml config settings.

- To set environment variables, use the `export VARNAME=value` command.
- To permanently set environment variables, add the export line to a file such as `~/.bashrc` or `~/.bash_profile`.

You can override any setting from a system property or in the `newrelic.yml` by setting an environment variable. The environment variable corresponding to a given setting in the config file is the setting name prefixed by `NEW_RELIC` with all dots (`.`) and dashes (`-`) replaced by underscores (`_`). For example, the environment variable for the `log_level` setting is `NEW_RELIC_LOG_LEVEL`.

For settings nested in stanzas, prepend the stanza name to the setting name. For example, the environment variable for the `enabled` setting in the `transaction_tracer` stanza is `NEW_RELIC_TRANSACTION_TRACER_ENABLED`.

Agent configuration via environment variables requires Java agent version 4.10.0 or higher.

For agent versions older than 4.10.0 the following environment variables are available:

Hide All

NEW_RELIC_APP_NAME (REQUIRED)

Type	String
Default	(none)

This setting is **required**. Contains the application name under which to report data to New Relic. Set the name of your application as you want it to appear in New Relic.

If `enable_auto_app_naming` is false, the agent reports all data to this application. Otherwise, the agent reports only background tasks (transactions for non-web applications) to this application.

To report data to more than one application, separate the application names with a semicolon `;`. For example, to report data to **My Application** and **My Application 2**:

```
app_name: My Application;My Application 2
```

For more methods of naming your application, see Name your Java application.

NEW_RELIC_DISTRIBUTED_TRACING_ENABLED

Type	String
Default	false

Enables distributed tracing. Case sensitive: use `true` or `false`. For more information, see the distributed tracing section.

NEW_RELIC_PROCESS_HOST_DISPLAY_NAME

Type	String
Default	(none)

Set a display name to decorate the "host:port" label in the New Relic UI.

NEW_RELIC_LICENSE_KEY (REQUIRED)

Type	String
Default	(none)

This setting is **required**. Contains your New Relic account license.

You must specify the license key associated with your New Relic account. This key binds your agent's data to your account in the New Relic service.

NEW_RELIC_LOG

Type	String
Default	`newrelic_agent.log`

The unqualified log file name or the string `STDOUT` which will log to standard out.

Cloud platform utilization

These options are set in the `utilization` stanza and can be overridden by using a `newrelic.config.utilization` prefixed system property.

The agent collects utilization information and sends it to the New Relic service. The agent can collect information from Amazon Web Services (AWS) EC2 instances and Docker containers.

Hide All

detect_aws

Type	Boolean

Default	`true`

Determines whether the agent polls AWS metadata API.

detect_docker

Type	Boolean
Default	`true`

Determines whether the agent reads Docker information from the file system.

Async instrumentation

These options are set directly in the `common` stanza and can be overridden by using a prefixed system property.

Hide All

token_timeout

Type	Integer
Default (seconds)	`180`

The number of seconds after which the agent will automatically expire an async token that has not been explicitly expired with `token.expire()`. For usage instructions, see Tokens: Connect async threads.

Increasing this value may impact performance, because it increases the amount of memory the agent uses and prevents transactions from being reported due to unexpired tokens.

segment_timeout

Type	Integer
Default (seconds)	`600`

The number of seconds after which the agent will automatically end a segment that has not been explicitly ended with `segment.end()` or `segment.ignore()`. For usage instructions, see Segments: Time arbitrary async activity.

Increasing this value may impact performance, because it increases the amount of memory the agent uses and prevents transactions from being reported due to un-ended segments.

Circuit breaker

These settings customize the behavior of the Java circuit breaker. These settings are not included in `newrelic.yml` by default. You do not need to restart your JVM after changing them.

If you want to customize the circuit breaker, add the stanza under the `common` stanza:

```
    common: &default_settings
      OTHER_CONFIG_SETTINGS
      circuitbreaker:
        enabled: true
        memory_threshold: 20
        gc_cpu_threshold: 10
```

Hide All

enabled

Type	Boolean
Default	`true`

If your application is behaving as expected, you may want to disable the circuit breaker.

memory_threshold

Type	Integer (0 to 100)
Default	20

Customize the precentage of free heap memory below which the circuit breaker should trip. When the percentage of free heap memory is less than `memory_threshold`, and the CPU time spent doing garbage collection is greater than `gc_cpu_threshold`, the circuit breaker trips. In order to make the circuit breaker less likely to trip, decrease `memory_threshold` and/or increase `gc_cpu_threshold`. Adjust these values as needed, based on your application's operating performance and behavior.

gc_cpu_threshold

Type	Integer (0 to 100)
Default	10

Customize the precentage of garbage collection CPU time above which the circuit breaker should trip. When the percentage of free heap memory is less than `memory_threshold`, and the CPU time spent doing garbage collection is greater than `gc_cpu_threshold`, the circuit breaker trips. In order to make the circuit breaker less likely to trip, decrease `memory_threshold` and/or increase `gc_cpu_threshold`. Adjust these values as needed, based on your application's operating performance and behavior.

Message tracer

These options are set in the `message_tracer` stanza and can be overridden by using a `newrelic.config.message_tracer` prefixed system property.

Hide All

segment_parameters.enabled

Type	Boolean
Default	`true`

Adds message properties to tracer attributes. Set this to `false` to turn it off.

Distributed tracing

Enabling distributed tracing disables cross application tracing, and has other effects on APM features. Before enabling, read the transition guide.

Requires Java agent version 4.3.0 or higher.

Distributed tracing lets you see the path that a request takes as it travels through a distributed system. In the config file, it can be set in the `distributed_tracing` stanza. It can be overridden using a `newrelic.config.distributed_tracing` prefixed system property.

Hide All

enabled

Type	Boolean
Default	`false`

Set this to `true` to enable distributed tracing.

For example, to enable this in the config file, you would use:

```
distributed_tracing:
   enabled: true
```

To enable this using a system property, you would use:

```
-Dnewrelic.config.distributed_tracing.enabled=true
```

exclude_newrelic_header

Type	Boolean
Default	`false`

By default, supported versions of the agent utilize both the `newrelic` header and W3C Trace Context headers for distributed tracing. The `newrelic` distributed tracing header allows interoperability with older agents that don't support W3C Trace Context headers. Agent versions that support W3C Trace Context headers will prioritize them over `newrelic` headers for distributed tracing.

If you do not want to utilize the `newrelic` header, setting this to `true` will result in the agent excluding the `newrelic` header and only using W3C Trace Context headers for distributed tracing.

For example, to exclude `newrelic` headers in the config file, you would use:

```
distributed_tracing:
   exclude_newrelic_header: true
```

To exclude `newrelic` headers using a system property, you would use:

```
-Dnewrelic.config.distributed_tracing.exclude_newrelic_header=true
```

Infinite Tracing

Requirements:

- Java Agent 5.12.1 or higher.
- Infinite Tracing does not work if `enable_auto_app_naming` is enabled.

To turn on Infinite Tracing, enable distributed tracing and add the additional setting below. For an example, see Language Agents: Configure Distributed Tracing.

Hide All

trace_observer.host

Type	String
Default	None

For help getting a valid Infinite Tracing trace observer host entry, see find or create a Trace Observer.

You can configure this via YAML:

```
infinite_tracing:
  trace_observer:
    host: YOUR_TRACE_OBSERVER_HOST
```

You can also use the system property `newrelic.config.infinite_tracing.trace_observer.host` or the environment variable `NEW_RELIC_INFINITE_TRACING_TRACE_OBSERVER_HOST`.

Span events

Span events are reported for distributed tracing. Distributed tracing must be enabled to report span events.

Span configuration is set in the `span_events` stanza and can be overridden by using a `newrelic.config.span_events` prefixed system property. Options include:

Hide All

enabled

Type	Boolean
Default	`true`

Used to enable/disable span event reporting.

attributes.enabled

Type	Boolean
Default	`true`

This setting can be used to turn on or off all attributes for span events. If `attributes.enabled` at the root level is `false`, no attributes will be sent to span events regardless on how this property (`span_events.attributes.enabled`) is set.

attributes.include

Type	List of strings
Default	(none)

If attributes are enabled for span events, all attribute keys found in this list will be sent to New Relic in `span_events`. For more information, see the agent attribute rules.

attributes.exclude

Type	List of strings
Default	(none)

All attribute keys found in this list will not be sent to New Relic in span events. For more information, see the agent attribute rules.

Span event attribute filtering requires Java agent version 4.10.0 or higher.

Jar collector

The Java agent collects and information about jars and their versions on the application classpath.

Jar collection configuration is set in the `jar_collector` stanza and can be overridden by using a `newrelic.config.jar_collector` prefixed system property. Options include:

Hide All

enabled

Type	Boolean
Default	`true`

Used to enable/disable jar collection and reporting.

skip_temp_jars

Type	Boolean
Default	`true`

Used to enable/disable collection of temporary jars. Temporary jars are those residing in the directory specified by the system property `java.io.tmpdir`.

jars_per_second

Type	Integer
Default	`10`

The maximum number of jars to process per second. Must be positive.

For more help

If you need more help, check out these support and learning resources:

- Browse the Explorers Hub to get help from the community and join in discussions.
- Find answers on our sites and learn how to use our support portal.
- Run New Relic Diagnostics, our troubleshooting tool for Linux, Windows, and macOS.
- Review New Relic's data security and licenses documentation.