Creatio Academy

Logging

Version 8.0

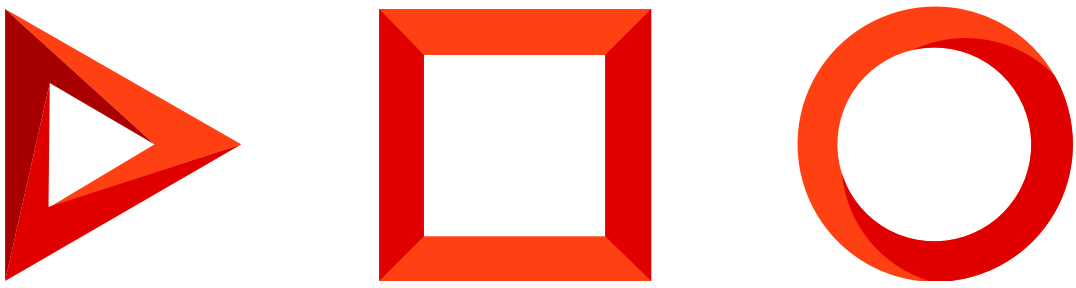


Table of Contents

NLog

Medium

We recommend enabling logging to verify that the new functionality operates as expected. For optimal performance, enable logging only while testing and debugging Creatio. Creatio can log all primary operations. This is achieved using the NLog library, a free .NET logging library that supports routing features and log management. NLog is suitable for any Creatio instance regardless of size or complexity.

The library lets you perform the following **actions**:

- handle diagnostic messages sent in any .NET language
- enrich logs with contextual data
- format logs based on user preferences
- send logs to one or more message receivers, such as a file or database

Learn more about the NLog on the GitHub website.

Set up logging for Creatio on-site

Creatio logs events for the loader and `Default` configuration separately.

You can set up logging for Creatio on-site in the following **ways**:

- via the configuration file
- via the `LoggingConfiguration` configuration object

View an example that sets up logging in a separate article: Implement a modal box.

Set up logging for Creatio on-site via the configuration file

1. Set the path to the log configuration file

You can set up logging in the following configuration **files** of the `..\Terrasoft.WebApp` directory:

- `nlog.config`
- `nlog.targets.config`

Set the path to the `nlog.config` file in the `..\Terrasoft.WebApp\Web.config` configuration file.

Example of the `nlog.config` file

```
<common>
  <logging>
    <factoryAdapter type="Common.Logging.NLog.NLogLoggerFactoryAdapter, Common.Logging.NLog45">
```

```
      <arg key="configType" value="FILE" />
      <arg key="configFile" value="~/nlog.config" />
    </factoryAdapter>
  </logging>
</common>
```

2. Specify the log receivers

Log receivers display, store and transfer the log messages to other receivers.

Log receivers have the following **types**:

- Receivers that receive and handle messages
- receivers that buffer or forward messages to another receiver

Specify the log receivers in the `<target>` XML element of the `...\Terrasoft.WebApp\nlog.targets.config` file.

The configuration **attributes** of log receivers are as follows:

- `name`. Receiver name.
- `xsi:type`. Receiver type. Available values: "`File`," "`Database`," "`Mail`."
- `fileName`. Log file and path to the log file. The log file location depends on the values of Windows system variables.

  Default path to the application loader log files

  ```
  [TEMP]\Creatio\Site_[SiteId]\[ApplicationName]\Log\[DateTime.Today]
  ```

  Path example

  ```
  C:\Windows\Temp\Creatio\Site_1\creatio806\Log\2022_05_22
  ```

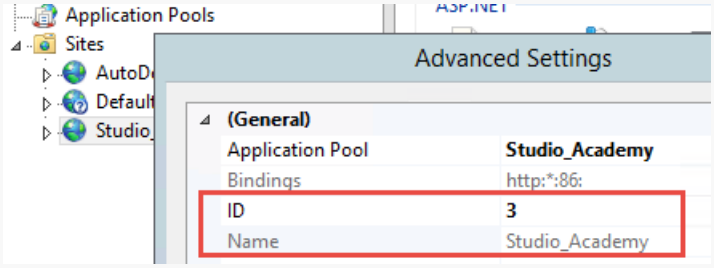
  Path to the Default configuration log files

  ```
  [TEMP]\Creatio\Site_[SiteId]\[ApplicationName]\[ConfigurationNumber]\Log\[DateTime.Today]
  ```

  Path example

  ```
  C:\Windows\Temp\Creatio\Site_1\creatio806\0\Log\2022_05_22
  ```

- `[TEMP]`. Base directory. By default, IIS uses the `C:\Windows\Temp` directory. Visual Studio (IIS Express) uses the `C:\Users\[User name]\AppData\Local\Temp` directory.
- `[SiteId]`. Website number. For IIS, find the website number in the advanced website settings. For Visual Studio, the number is 2.
- `[ApplicationName]`. Creatio name.
- `[ConfigurationNumber]`. Configuration number. The number for the `Default` configuration is usually 0.
- `[DateTime.Today]`. Log date.



- `layout`. Template for populating the log file.

Example of the `nlog.targets.config` file

```
<target name="universalAppender" xsi:type="File"
    layout="${DefaultLayout}"
    fileName="${LogDir}${LogDay}${logger:shortName=True}.log" />
```

Learn more about log receivers in the official NLog documentation.

3. Define the log rules

You can define the log rules in the `nlog.config` file.

The configuration **attributes** of log rules are as follows:

- `name`. Log name.
- `minlevel`. Minimum logging level. The default logging level for Creatio components is set to ensure the highest performance.

  Available **logging levels** in ascending priority:

  - `Trace`. Log all events during setup. Set the initial and final methods.
  - `Debug`. Log all events during debugging.
  - `Info`. Log regular Creatio activity.
  - `Warn`. Log warnings. Creatio continues to operate after logging.
  - `Error`. Log errors that might crash Creatio.
  - `Fatal`. Log errors that inevitably crash Creatio.

  - `Off`. Disable logging.
- `maxlevel`. Maximum logging level.
- `level`. Log events of a specific logging level.
- `levels`. Log events of multiple logging levels, separated by commas.
- `writeTo`. The name of the log receiver.
- `final`. Whether to handle subsequent rules.
- `enabled`. Disable the log rule (set to `false`) without deleting it.
- `ruleName`. The log rule ID.

NLog handles logging rule attributes in the following **order**:

1. `level`
2. `levels`
3. `minlevel` and `maxlevel`

If `minLevel = "Warn" level = "Info"`, only the `Info` logging level is used, because the handling priority of the `level` attribute is higher than `minLevel`.

Example of a log rule that writes a log to the database

```
<logger name="IncidentRegistration" writeTo="AdoNetBufferedAppender" minlevel="Trace" final="tru
```

Set up logging for Creatio on-site via the LoggingConfiguration configuration object

1. Create a `LoggingConfiguration` object that describes the configuration.
2. Create the log receivers.
3. Set up the properties of the log receivers.
4. Define the logging rules using `LoggingRule` objects.
5. Add the logging rules to the `LoggingRules` configuration objects.
6. Activate the configuration. To do this, specify the created configuration object in `LogManager.Configuration`.

Set up logging for Creatio in the cloud

To **set up logging for Creatio in the cloud**, contact Creatio support.

Attention. It is impossible to add an additional log receiver when setting up logging for Creatio in the cloud.

Set up logging of incoming HTTP requests in Creatio for .NET Core and .NET 6

Beginner

You can set up logging of incoming HTTP requests in Creatio **.NET Core** version 7.17.3-8.0.7 and Creatio **.NET 6** version 8.0.8 and later.

Creatio lets you use the following logging **types** of incoming HTTP requests:

- standard logging
- extended logging

Standard logging of incoming HTTP requests

Standard logging lets you log incoming HTTP requests. The `Request.log` file in the `Logs` directory of the Creatio root directory stores standard logs.

To **set up standard logging**:

1. Open the `appsettings.json` configuration file in the Creatio root directory.
2. Move to the `Standard` block. Standard logging is enabled by default (the `Enabled` flag is set to `true`).
3. Add HTTP response codes into the `StatusCodes` flag. Creatio will log the requests that receive the specified response codes. If you turn on the `Enabled` flag and leave the `StatusCodes` flag empty, Creatio logs all incoming HTTP requests.

Example that configures standard logging (the `appsettings.json` file)

```
"RequestLogging": {
    "Standard": {
        "Enabled": true,
        "StatusCodes": [ 200, 300, 302 ]
    },
    ...
}
```

You can analyze the logs using the Log Parser Studio utility because the log format is similar to the IIS log format.

Extended logging for incoming HTTP requests

Extended logging lets you retrieve detailed information about logs for incoming HTTP requests. The `ExtendedRequest.log` file in the `Logs` directory of the Creatio root directory stores extended logs. Creatio saves data about the response body of an incoming HTTP request automatically when extended logging is turned on.

To **set up extended logging**:

1. Open the `appsettings.json` configuration file in the Creatio root directory.
2. Move to the `Extended` block. Extended logging is turned off by default (the `Enabled` flag is set to `false`).
3. Set the `Enabled` flag to `true` to turn on logging.
4. If you want to retrieve information about the HTTP request body in the log, set the `LogRequestBody` flag to `true` (`false` by default).
5. Set the size of the displayed request/response body (the first N bytes) in the `MaxBodySizeBytes` flag.

> **Attention.** Extended logging affects performance if Creatio receives a large number of requests or you set a large value of the `MaxBodySizeBytes` flag.

6. Add HTTP response codes into the `StatusCodes` flag. Creatio will log the requests that receive the specified response codes. If you turn on the `Enabled` flag and leave the `StatusCodes` flag empty, Creatio logs all incoming HTTP requests.

Example that configures extended logging (the `appsettings.json` file)

```
"RequestLogging": {
    ...,
    "Extended": {
        "Enabled": true,
        "LogRequestBody": false,
        "MaxBodySizeBytes": 500,
        "StatusCodes": [ 400, 401, 403 ]
    }
}
```

You can use standard and extended logging simultaneously with different values of the `StatusCodes` flag. If the `StatusCodes` flag values for standard and extended logging match, Creatio duplicates the logs for incoming HTTP requests in the `Request.log` and `ExtendedRequest.log` files with different details.