On this page, you can configure settings of the system's debugging tools. The following categories of settings are available:

- General
- All
- Cache access
- SQL queries
- IO
- Page ViewState
- Output
- Security
- Macros
- Analytics
- Requests
- Web farm
- Event handlers

General

Setting	Description
Disable debugging	Globally disables all debugs, regardless of individual debug settings.
Debug Import /Export	If disabled, the system does not log debug information for the pages of the Import/Export interface. For performance reasons, it is recommended to leave this option disabled unless you need to debug the Import/Export process.
Debug resources	If disabled, the debug ignores all resource requests (*GetResource* and *GetCSS*).
Debug scheduler	If disabled, the debug ignores all operations executed by the scheduler.

All

Setting	Description
Debug everything everywhere	Enables: - All debug types - Debugging of user interface pages for all debugs - Live site debugging (for all debugs)
Enable all debugs	Enables all debugs and the corresponding tabs in the **Debug** application.
Display all debugs on live site	If checked (true), all *enabled* debugs display their information on the live site below the regular content of pages.
Include UI pages in all debugs	If checked (true), all *enabled* debugs include actions performed on the pages of the user interface.
Default log length	Sets the default maximum length of the debug log in the **Debug** application. The default log length is used by debug types that do not have their own log length set.

Display stack information in every debug	If enabled, the system tracks the code stack for all debug types and displays the information in the Context column of the debug interface in the **Debug** application.
Log everything to file	Enables logging of all possible operations into *.log* files in the **~/App_Data/** folder (including the Event log and E-mail sending log).

Cache access

Setting	Description
Enable cache access debug	Enables cache access debugging and the **Cache access** tab in the **Debug** application.
Display cache access debug on live site	If enabled, cache access debug information is also displayed at the bottom of each live site page. This option requires cache access debugging to be enabled.
Debug cache access of UI pages	If enabled, access to data cached for pages of the administration interface will also be included in the cache access debug. This option requires cache access debugging to be enabled.
Cache access debug log length	Sets the maximum length of the cache access debug log on the **Debug -> Cache access** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging cache access and displays the information in the **Context** column. This information is only available in the debugging UI and on the live site, not in the debug log written into the *log cache.log* file.
Log cache access to file	If enabled, cache access debug log is saved into the *logcache.log* file in the *~\App_Data* folder. This option does not require cache access debugging to be enabled.

SQL queries

Setting	Description
Enable SQL query debug	Enables SQL query debugging and the **SQL queries** tab in the **Debug** application.
Display SQL query debug on live site	If enabled, SQL query debug information is also displayed at the bottom of each live site page. This option requires SQL query debugging to be enabled.
Debug SQL queries of UI pages	If enabled, SQL queries called by pages of the administration interface will also be included in the SQL query debug. This option requires SQL query debugging to be enabled.
SQL query debug log length	Sets the maximum length of the SQL query debug log on the **Debug -> SQL queries** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Debug SQL connections	If enabled, SQL connection operations (new, open, close) are logged in the SQL query debug log.

Display stack information	If enabled, the system tracks the code stack when debugging SQL queries and displays the information in the **Cont ext** column. This information is only available in the debugging UI and on the live site, not in the debug log written into the *logs ql.log* file.
Log SQL queries to file	If enabled, SQL query debug log is saved into the *logsql.log* file in the *~\App_Data* folder. This option does not require SQL query debugging to be enabled.

IO

Setting	Description
Enable IO operation debug	Enables IO operation debugging and the **IO** tab in the **Debug** application.
Display IO operation debug on live site	If enabled, IO operation debug information is also displayed at the bottom of each live site page. This option requires IO operation debugging to be enabled.
Debug IO operations of UI pages	If enabled, IO operations called by pages of the administration interface will also be included in the IO operation debug. This option requires IO operation debugging to be enabled.
IO operation debug log length	Sets the maximum length of the IO operation debug log on the **Debug -> IO** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging IO operations and displays the information in the **C ontext** column. This information is only available in the debugging UI and on the live site, not in the debug log written into the *log files.log* file.
Log IO operations to file	If enabled, IO operation debug log is saved into the *logfiles.log* file in the *~\App_Data* folder. This option does not require IO operation debugging to be enabled.

Page ViewState

Setting	Description
Enable ViewState debug	Enables ViewState debugging and the **Page ViewState** tab in the **Debug** application.
Display ViewState debug on live site	If enabled, ViewState debug information is also displayed at the bottom of each live site page. This option requires ViewState debugging to be enabled.
Debug ViewState of UI pages	If enabled, ViewState of administration interface pages will also be included in the ViewState debug. This option requires ViewState debugging to be enabled.

ViewState debug log length	Sets the maximum length of the ViewState debug log on the **Debug -> ViewState** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength**) key) is used.

Output

Setting	Description
Enable output debug	Enables page output debugging and the **Output** tab in the **Debug** application.
Debug output of UI pages	If enabled, output of administration interface pages will also be included in the output debug. This option requires output debugging to be enabled.
Output debug log length	Sets the maximum length of the output debug log on the **Debug -> Output** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMS DebugEverythingLogLength** key) is used.
Log output to file	If enabled, output debug log is saved into the *logOutput.log* file in the ~*App_Data* folder. This option does not require output debugging to be enabled.

Security

Setting	Description
Enable security debug	Enables security operation debugging and the **Security** tab in the **Debug** application.
Display security debug on live site	If enabled, security operation debug information is also displayed at the bottom of each live site page. This option requires security debugging to be enabled.
Debug security operations of UI pages	If enabled, security checks performed by pages of the administration interface will also be included in the security debug. This option requires security debugging to be enabled.
Security debug log length	Sets the maximum length of the security debug log on the **Debug -> Security** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging security and displays the information in the **Context** column. This information is only available in the debugging UI and on the live site, not in the debug log written into the *log Security.log* file.
Log security operations to file	If enabled, security debug log is saved into the *logSecurity.log* file in the ~*App_Data* folder. This option does not require security debugging to be enabled.

Macros

Setting	Description
Enable macro debug	Enables macro debugging and the **Macros** tab in the **Debug** application.
Enable detailed macro debug	If enabled, the macro debug displays the results of all subelements used within macro expressions. This allows you to check the exact data content of a macro's components during each step of the resolving process. Detailed macro debugging is highly recommended if you encounter problems with complex expressions. If disabled, the debug only shows the final result of each macro. You can enable the detailed debug only for specific expressions by adding the **|(debug)** parameter to the given macro.
Display macro debug on live site	If enabled, macro debug information is also displayed at the bottom of each live site page. Macro debugging must also be enabled.
Debug macros resolved on UI pages	If enabled, macros resolved on the pages of the administration interface are also included in the macro debug. Macro debugging must also be enabled.
Macro debug log length	Sets the maximum length of the macro debug log, i.e. the number of requests for which the macro debug stores information. If empty, the value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging macros and displays the information in the **Context** column. This information is only available in the debugging UI and on the live site, not in the debug log written into the *logmacros.log* file.
Log macros to file	If enabled, the system saves the macro debug log into the *logmacros.log* file in the *~\App_Data* folder. Macro debugging must also be enabled.

Analytics

Setting	Description
Enable web analytics debug	Enables web analytics debugging and the **Analytics** tab in the **Debug** application.
Display web analytics debug on live site	If enabled, web analytics debug information is also displayed at the bottom of each page on the live site. Requires web analytics debugging to be enabled. Debugging on the live site is only available if your site does NOT use [JavaScript logging for web analytics](#) (**Online marketing -> Web analytics -> Log via JavaScript snippet** setting).
Web analytics debug log length	Sets the maximum length of the web analytics debug log on the **Debug -> Analytics** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.

Display stack information	If enabled, the system tracks the code stack when debugging web analytics and displays the information in the **Context** column.
	This information is only available in the debugging UI and on the live site, not in the debug log written into the *log analytics.log* file.
Log web analytics to file	If enabled, web analytics debug log is saved into the *loganalytics.log* file in the *~\App_Data* folder. This option does not require web analytics debugging to be enabled.

Requests

Setting	Description
Enable request debug	Enables request debugging and the **Requests** tab in the **Debug** application.
Display request debug on live site	If enabled, request debug information is also displayed at the bottom of each live site page. This option requires request debugging to be enabled.
Debug UI page requests	If enabled, administration interface page requests will also be included in the macro debug. This option requires request debugging to be enabled.
Request debug log length	Sets the maximum length of the request debug log on the **Debug -> Requests** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging requests and displays the information in the **Context** column.
	This information is only available in the debugging UI and on the live site, not in the debug log written into the *logRequests.log* file.
Log requests to file	If enabled, request debug log is saved into the *logRequests.log* and *logRequestsUrls.log* files in the *~\App_Data* folder. The first file contains the full log, while the second one only lists requested URLs. This option does not require request debugging to be enabled.

Web farm

Setting	Description
Enable web farm debug	Enables request debugging and the **Web farm** tab in the **Debug** application.
Debug web farm operations of UI pages	If enabled, operations performed via the administration interface will also be included in the web farm debug. This option requires web farm debugging to be enabled.
Web farm debug log length	Sets the maximum length of the web farm debug log on the **Debug -> Requests** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.

Display stack information	If enabled, the system tracks the code stack when debugging web farm operations and displays the information in the **Context** column.
	This information is only available in the debugging UI and on the live site, not in the debug log written into the *log webfarm.log* file.
Log web farm operations to file	If enabled, web farm debug log is saved into the *logwebfarm.log* file in the *~\App_Data* folder. This option does not require web farm debugging to be enabled.

Event handlers

Setting	Description
Enable handlers debug	Enables event handlers debugging and the **Event handlers** tab in the **Debug** application.
Display handlers debug on live site	If enabled, event handlers debug information is also displayed at the bottom of each live site page. This option requires event handlers debugging to be enabled.
Debug handlers in UI pages	If enabled, operations performed via the administration interface will also be included in the event handlers debug. This option requires event handlers debugging to be enabled.
Handlers debug log length	Sets the maximum length of the event handlers debug log on the **Debug -> Event handlers** tab, i.e. the number of requests for which debug information is preserved and displayed on the tab. If empty, value of the **Default log length** setting (or the **CMSDebugEverythingLogLength** key) is used.
Display stack information	If enabled, the system tracks the code stack when debugging event handlers and displays the information in the **Context** column.
	This information is only available in the debugging UI and on the live site, not in the debug log written into the *logwebfarm.log* file.
Log handlers to file	If enabled, event handlers debug log is saved into the *logwHandlers.log* file in the *~\App_Data* folder. This option does not require event handlers debugging to be enabled.