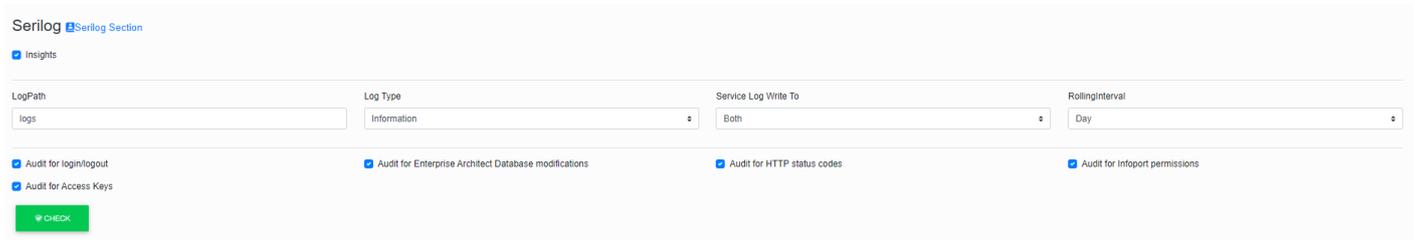


Serilog

The third section allows us to set the Infoport logging.



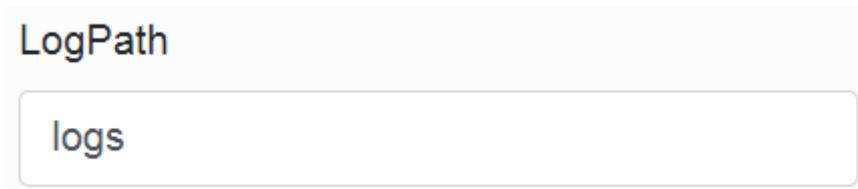
The screenshot shows the Serilog configuration page. At the top, there's a header "Serilog" with a link to the "Serilog Section". Below it, a checkbox labeled "Insights" is checked. The main configuration area contains four input fields: "LogPath" with the value "logs", "Log Type" with a dropdown menu showing "Information", "Service Log Write To" with a dropdown menu showing "Both", and "RollingInterval" with a dropdown menu showing "Day". Below these fields, there are four audit options, each with a checked checkbox: "Audit for login/logout", "Audit for Access Keys", "Audit for Enterprise Architect Database modifications", and "Audit for HTTP status codes". The last option, "Audit for Infoport permissions", is also checked. At the bottom left, there is a green "CHECK" button.

The first item is a check box that says whether user activities should be logged. (List of visited URLs).



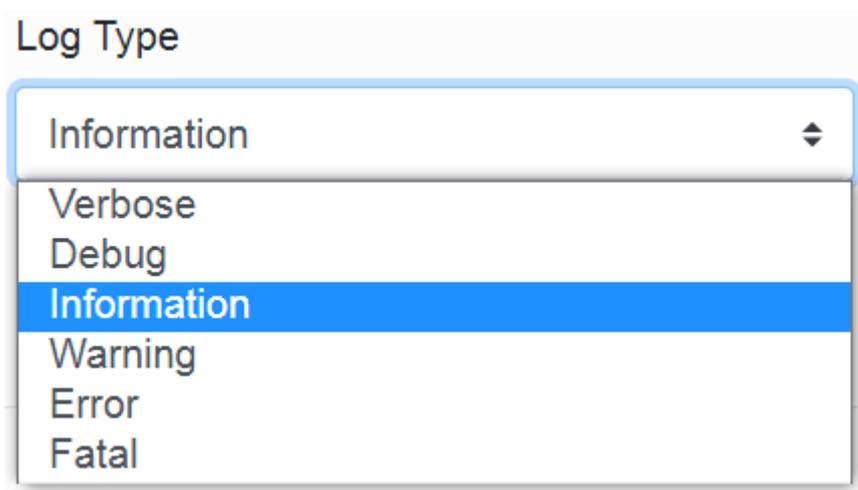
A close-up of the "Insights" checkbox, which is checked and highlighted with a blue background.

In the second item, we choose the relative path for saving logs.



A close-up of the "LogPath" input field, which contains the text "logs".

In the third item, we select the type of logging.
(Each type is described in the table. We recommend Information logging.)



A close-up of the "Log Type" dropdown menu. The menu is open, showing a list of logging levels: Verbose, Debug, Information (highlighted in blue), Warning, Error, and Fatal. The "Information" option is selected.

Table for logging types.

Level (from the most detailed to the least detailed)	Description
Verbose	For information that's typically valuable only for debugging. These messages may contain sensitive application data and so shouldn't be enabled in a production environment. Disabled by default.
Debug	For information that may be useful in development and debugging. Example: Entering method Configure with flag set to true. Enable Debug level logs in production only when troubleshooting, due to the high volume of logs.
Information	For tracking the general flow of the app. These logs typically have some long-term value. Example: Request received for path/api/todo
Warning	For abnormal or unexpected events in the app flow. These may include errors or other conditions that don't cause the app to stop but might need to be investigated. Handled exceptions are a common place to use the Warning log level. Example: FileNotFoundException for file quotes.txt.
Error	For errors and exceptions that cannot be handled. These messages indicate a failure in the current activity or operation (such as the current http request), not an app-wide failure. Example log message: Cannot insert record due to duplicate key violation.
Fatal	For failures that require immediate attention. Examples: data loss scenarios, out of disk space.

In the fourth item, we select where we want the logs to be written. We have three options: Console, File or Both.

Service Log Write To

Both ▾

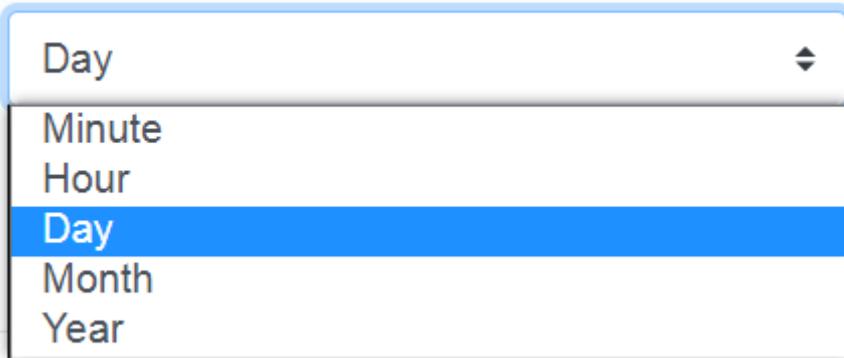
Console

File

Both

In the fifth item, we choose how often the log file should be closed.

RollingInterval



A dropdown menu titled "RollingInterval" is shown. The menu is open, displaying a list of options: "Day", "Minute", "Hour", "Day", "Month", and "Year". The "Day" option is currently selected and highlighted with a blue background. A small downward-pointing arrow is visible on the right side of the dropdown box.

Here we can see the chosen day. This means that a new log file is created for the portal every day. Logs from previous days remain on disk.



After filling in, just press the button and the manager will tell you if everything is OK.

Revision #4

Created 7 April 2022 11:18:40

Updated 23 April 2024 12:48:20 by Karolína Kavanová