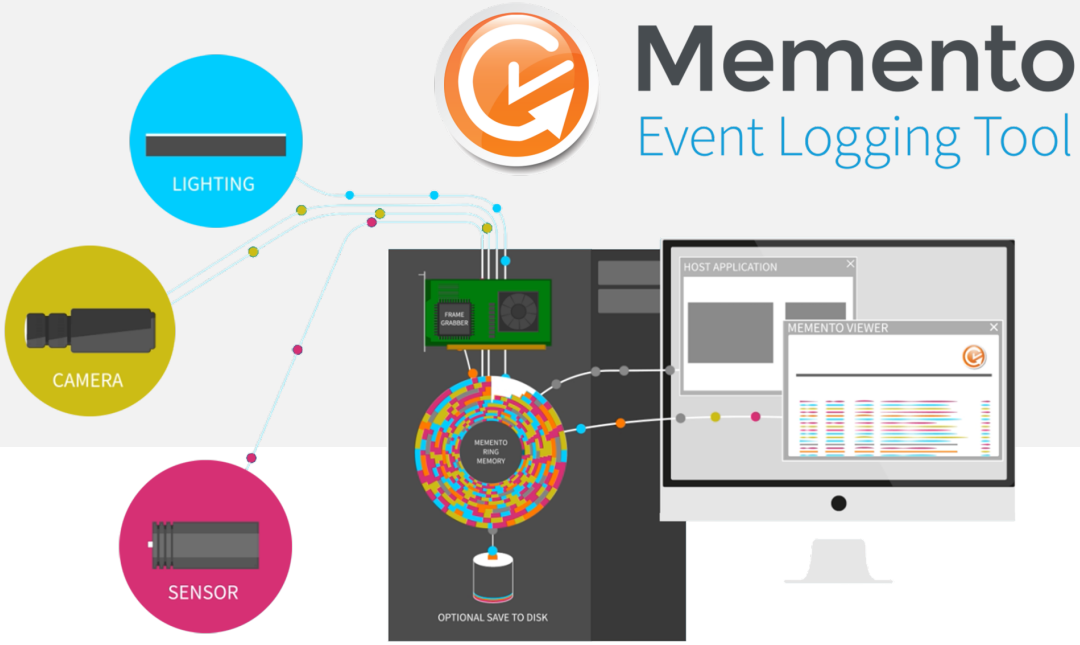
EURESYS™
Excellence in vision

GETTING STARTED

Memento

Memento 9.4.0



Terms of Use

EURESYS s.a. shall retain all property rights, title and interest of the documentation of the hardware and the software, and of the trademarks of EURESYS s.a.

All the names of companies and products mentioned in the documentation may be the trademarks of their respective owners.

The licensing, use, leasing, loaning, translation, reproduction, copying or modification of the hardware or the software, brands or documentation of EURESYS s.a. contained in this book, is not allowed without prior notice.

EURESYS s.a. may modify the product specification or change the information given in this documentation at any time, at its discretion, and without prior notice.

EURESYS s.a. shall not be liable for any loss of or damage to revenues, profits, goodwill, data, information systems or other special, incidental, indirect, consequential or punitive damages of any kind arising in connection with the use of the hardware or the software of EURESYS s.a. or resulting of omissions or errors in this documentation.

This documentation is provided with Memento 9.4.0 (doc build 6001).

Contents

1. Abstract

Memento Logo

Memento is an advanced event message logging system that greatly facilitates the debugging of machine vision applications using Euresys frame grabbers.

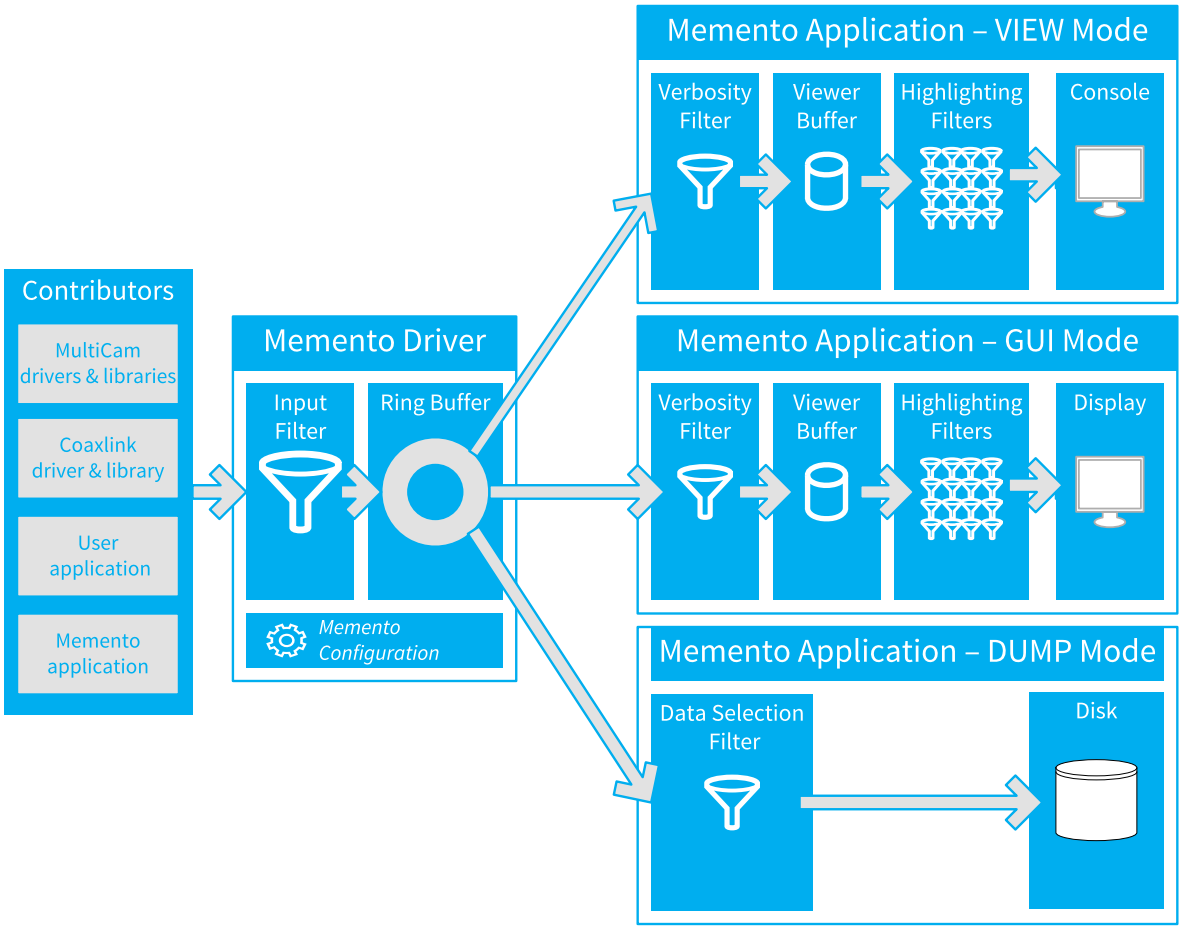
Memento is a set of software tools allowing:

- Kernel drivers and user space applications – **Memento Contributors** – to inject trace messages – **Memento Messages** – into a common memory area – the **Memento Ring Buffer**.
- Memento Contributors to time-stamp Memento messages using a common time scale – **Memento Time Scale**.
- To view selected sets of recent- or past- Memento Messages using the **Memento Viewer** function
- To dump Memento data from the Memento Ring Buffer to disk using the **Memento Dump** function

The Memento software package has two main components:

- A kernel-mode driver: **Memento Driver**.
- A user-mode application: **Memento Application**.

2. Block Diagram



Memento Function Blocks and Data Flow

3. Contributors

The Memento Contributors are Euresys kernel drivers and Euresys or third-party user space applications embedding **Memento Traces** at particular code locations.

A Memento Trace is a non-blocking and lightweight software macro that, upon execution, builds up a message and pushes it into the Memento Ring.

Coaxlink Driver & Library

The Coaxlink Driver software package is a major message contributor. For instance, Coaxlink Driver 4.7 includes ~1,000 distinct instances of Memento Traces!

All the Coaxlink products implement **Memento-in-hardware**. This feature allows hardware events to be logged in Memento with an accurate time attribute.

MultiCam Driver & Libraries

The kernel-mode drivers for Grablink Base, Grablink DualBase, Grablink Full, and Grablink Full XR products as well as the MultiCam and clseremc libraries can also contribute to Memento.

User Application

A user application may include Memento Traces capable of generating messages having a custom content and two attributes: `kind` and `level`.

Any severity level can be assigned to the `level` attribute.

Sixteen (16) distinct values are available for the `kind` attribute:`user0`, … `user9`, … `userA`, … `userF`.

> **Note:** *The value `user` is kept for backward capability; it is equivalent to `user0`.*

EGrabber

The EGrabber API layer generates Memento Traces with the `Egrabber` kind attribute.

Memento Application

The Memento Application also acts as a message contributor since it may inject "UTC Time" messages.

4. Messages

A Memento message is composed of a **message body** and several **message attributes**.

Message Body

The message body is a text string composed of a **fixed frame** and optional **arguments**. The arguments are used as placeholders for variable context information.

Message body examples:

- The following message body has no arguments:

```
GenICam module has been unloaded
```

- The following message body has two arguments, the date and the time:

```
Time: 2016-04-22 14:43:29.0493338 UTC
```

Message Attributes

Message attributes are used for qualifying the message body.

- Mandatory Attributes
  - The `Kind` and `Level` attributes are mandatory.
  - They must always be defined by the Memento Contributors.
- Optional Attributes
  - The `Time`, `Process ID`, `Thread ID`, `Card ID`, and `Connector ID` attributes are optional.
  - The Memento Contributors decide whether or not to use them.

Level Attribute

The `Level` attribute defines the **severity level** of the message on a scale having 7 steps:

Value	Severity	Description
Critical	Highest	Unrecoverable error message
Error	Higher	Error message
Warning	High	Warning message
Notice	Medium	Notification message
Info	Low	Information message

Value	Severity	Description
Debug	Lower	Debug message
Verbose	Lowest	Nattering

Note: *Debug and verbose levels are used for messages intended for Euresys engineering and technical support.*

Kind Attribute

The mandatory `Kind` attribute defines the kind, nature, origin … of the message:

Value	Description
API	API function calls.
Acquisition	Messages related to image acquisition (e.g., data stream start/stop, start/end of scan).
CoaXPress	Messages related to CoaXPress protocol (e.g., device discovery, control messages).
DMA	Messages related to DMA (Direct Memory Access) transfers.
DPC	Messages related to DPC (Deferred Procedure Calls) queued by the driver when it receives an interrupt request.
Descriptors	Messages related to DMA descriptors.
EEPROM	Messages related to on-board EEPROM (where part number, serial number and OEM key are stored).
Egrabber	Messages produced by EGrabber.
Event	Messages related to the event notification infrastructure.
FPGA	Messages related to : • Firmware image. • CIC (Camera and Illumination Controller).
Flash	Messages related to the on-board flash memory (where the firmware image is stored).
GenAPI	Messages related to Euresys GenApi implementation.
Hardware	Messages emitted by the frame grabber firmware.
I2C	Messages related to the I2C communication between 2 frame grabber components.
IOCTL	Messages related to the communication between user-mode software components and the kernel driver.
IRQ	Messages related to the interrupt requests received by the kernel driver.
LUT	Messages related to the look-up table processing.

Value	Description
Memento	Messages related to Memento.
OperationM	Messages related to an internal software component (should not appear on customers machines).
PCI Express	Messages related to the PCI Express configuration space (e.g., vendor and device IDs, capabilities).
PnP	Messages related to the interaction between the driver and the Windows Plug and Play Manager.
Power	Messages related to the interaction between the driver and the Windows Power Manager.
Profiling	A small number of messages for doing profiling (IRQ counter, time spent with interrupts masked, DPC counter, time spent in DPC, buffer push/pop and DMA transfer completion).
SPI	Messages related to the SPI interface (used to access the on-board flash).
Script	Messages related to the GenApi scripts.
Serial Line	Messages related to the Camera Link serial communication.
Thread	Messages related to worker threads used by the driver.
Time	Messages related to system data and time.
Timer	Messages related to operating system timers.
User0 … User9, UserA … UserF	Messages emitted by the user application.

Note: *The above list is open; new values can be appended in future Memento releases.*

Time Attribute

For all but the hardware kind of messages, the `time` attribute defines the time of occurrence on the **Memento Time Scale** of the creation of the message by the contributor.

For hardware kind of messages, the `time` attribute defines the time of occurrence on the **Memento Time Scale** of the hardware event that is at the origin of the message. These messages are prefixed by [ts:HARDWARE_TIMESTAMP] where HARDWARE_TIMESTAMP is the time of occurrence on the **card local time scale**.

Note: *The time of occurrence reported by the hardware using the **card local time** is converted to Memento time.*

Process ID & Thread ID Attributes

The `Process ID` attribute reports the process identifier of the process producing the message.

The `Thread ID` attribute reports the thread identifier of the thread producing the message.

Card ID & Connector ID Attributes

The `Card ID` attribute reports the identifier of the frame grabber card the message belongs to. This is a zero-based index assigned by the corresponding card driver.

- The `Card ID` is global within a specific driver.
- Grablink cards have a specific driver for each card type. If, for instance, a system has 2 Grablink Base and 2 Grablink DualBase, there will be Memento traces related to Grablink Base Card ID #0 and #1 as well as Grablink DualBase card ID #0 and #1.
- Coaxlink cards have a unique driver for all types. If, for instance, a system has 1 Coaxlink Mono and 1 Coaxlink Duo, there will be Memento traces related to Coaxlink Card ID #0 and #1.

The `Connector ID` attribute reports the identifier of the camera connector the message belongs to.

- For Coaxlink cards, the `Connector ID` is the ID (A, B, C, D) of the CoaXPress host interface connector.
- For Grablink cards, the `Connector ID` is the ID (M, A, B) of the Camera Link connector.

5. Time Scale

The Memento Time Scale is a common time scale used for time-stamping all the Memento Messages produced by the Memento Contributors within a Host PC.

Having a common time scale is a key feature of Memento since it allows messages from different origins to be time-coordinated!

The Memento Time Scale:

- Resets at the cold boot of the Host PC.
- Increments by 1 every microsecond.
- Never overflows thanks to a 64-bit quantification.

Memento uses the Performance Counter of the Host PC to build the Memento Time Scale. Consequently, its accuracy and its stability are Host PC dependent!

> **Note:** ***For Windows users only:*** *The Memento Time Scale is not reset when Windows performs a Fast Boot, even after a Power Off! When the Fast Boot option is activated, the only way to reset the Memento Time Scale, is to perform a Windows Restart.*

UTC vs. Memento Time

When a Coaxlink card is installed in the Host PC, the Coaxlink Driver generates every minute an UTC time message:

```
Time: 2016-04-22 14:43:29.0493338 UTC
```

Such message allows for establishing the relationship between the UTC (Coordinated Universal Time) and the Memento Time.

The time reported in the above message is actually the Host PC time of the local time zone converted to the UTC time zone.

> **Note:** *The accuracy of the reported UTC time depends on the Host PC. When the Host PC implements a synchronization mechanism to Internet time servers, the UTC time accuracy is pretty good!*

6. Driver

The kernel-mode Memento Driver is responsible for managing the **Memento Ring Buffer**: a region of the Host PC memory where the **Memento Contributors** store the **Memento Messages**.

The Coaxlink Driver and/or the MultiCam Driver automatically connects to the Memento Driver.

Once the connection is established, the messages composed by the contributors are first screened by the **Ring Buffer Filter** according to the **Ring Buffer Configuration** settings.

Only the messages that successfully pass through the Ring Buffer Filter filter are effectively written into the Memento Ring Buffer.

6.1. Ring Buffer

The Memento Ring Buffer is an area of the Host PC memory where the Memento messages are written by the contributors.

The Memento Ring Buffer:

- Has a fixed capacity of 524.288 (512 K) message slots of 64 bytes each.
- Is managed as a "ring buffer": A new message replaces the oldest one when it becomes full.

Message Compacting

To optimize the storage and the throughput efficiencies of the Memento Ring Buffer, Memento messages issued by Euresys Drivers or Libraries are stored in a compact form.

A compacted message without arguments occupies only a single 64-byte memory slot in the Ring Buffer. When the message has arguments, it requires additional memory slots.

The original messages are reconstructed by the Memento Viewer application.

Messages issued by user applications are not compacted and thus, require more space in the Memento Ring Buffer.

Sequential Number

The Memento Driver assigns a sequential number to every message entering the Memento Ring Buffer.

The sequential number is a 64-bit positive integer number that increments by 1 on every message input. The sequential number is not reset every boot session.

The sequential number unambiguously identifies a Memento message in the whole message history of a Host PC.

Ring Buffer Persistence (Windows only)

The Memento Driver maintains a synchronized copy of the Memento Ring Buffer on disk.

This allows the state and the content of the Memento Ring Buffer to be restored at the next boot session after a shutdown.

This mechanism operates correctly when the current session terminates normally.

> **Note:** *If the current session terminates with a "blue screen", or if the reset button is pushed, or if the power supply is removed, the Memento Ring Buffer data is appended to the* ***crash dump*** *file.*

6.2. Ring Buffer Filter

A message filter – named input filter – is inserted in the message flow entering the Memento Ring Buffer.

The input filter checks the `level` attribute of all the incoming messages against the **input filter rules** and decides if the message has to be logged or not into the Memento Ring Buffer.

Input Filter Rules

The input filter rules set is composed with:

- One **default rule**.
- Zero or more **kind-specific rules**.

Default Rule

The default rule has one argument: `level`.

The default rule applies only to messages having a `kind` attribute that is NOT matching any kind-specific rule.

Kind-specific Rules

A kind-specific rule has two arguments: `kind` and `level`.

The rule applies only to messages having the specified `kind` attribute.

The kind-specific rules supersede the default rule.

Input Filter Action

Are logged into the Memento Ring Buffer:

- Any message having no matching kind-specific rule and having a severity level greater than or equal to the specified level in the default rule.

- Any message having a matching kind-specific rule and having a severity level greater than or equal to the specified level in the matching kind-specific rule.

The other messages are rejected and lost forever.

Level Filtering

Rule Level	**Accepted Message Levels**
Critical	Critical
Error	Critical Error
Warning	Critical Error Warning
Notice	Critical Error Warning Notice
Info	Critical Error Warning Notice Info
Debug	Critical Error Warning Notice Info Debug
Verbose	Critical Error Warning Notice Info Debug Verbose

6.3. Ring Buffer Configuration

The Memento Configuration defines the input filter rules.

Memento Configuration Changes

The Memento Configuration settings can be changed at any time and take effect immediately.

Memento Configuration Persistence (Windows only)

The Memento Configuration settings are persistent. The settings are stored into a configuration file on disk and are automatically restored at the next boot session.

Default Memento Configuration

The default Memento configuration defines a default rule for the input filter with the level argument set to `info`. This rule keeps only the messages of any kind having a verbosity level `info` or higher.

7. Application Interfaces

Memento is a multi-mode application. Possible modes are:

Short Name	**Full Name & Description**
`gui`	**Graphical User Interface** (Windows default mode) A single-window graphical user interface application to: • Show/Change the Memento Configuration • Monitor graphically the input activity of the Memento Ring Buffer • View a selected set of messages of the Memento Ring Buffer • Control interactively the selection of messages to display • Dump the Memento Ring Buffer data to disk *Windows default mode.*
`config`	**Console Configurator** A command prompt console application to: • Show/Change the Memento Configuration
`view`	**Console Viewer** A command prompt console application to: • View a selected set of messages of the Memento Ring Buffer *Linux default mode.*
`dump`	**Console Dumper** A command prompt console application to: • Dump the Memento Ring Buffer data to disk

Multiple instances of the Memento Application can run concurrently!

8. Viewer

Memento messages can be retrieved selectively from the Memento Ring Buffer and viewed using the **Memento Viewer** function of the Memento Application.

The Memento Viewer function is available in the following two modes of Memento Application:

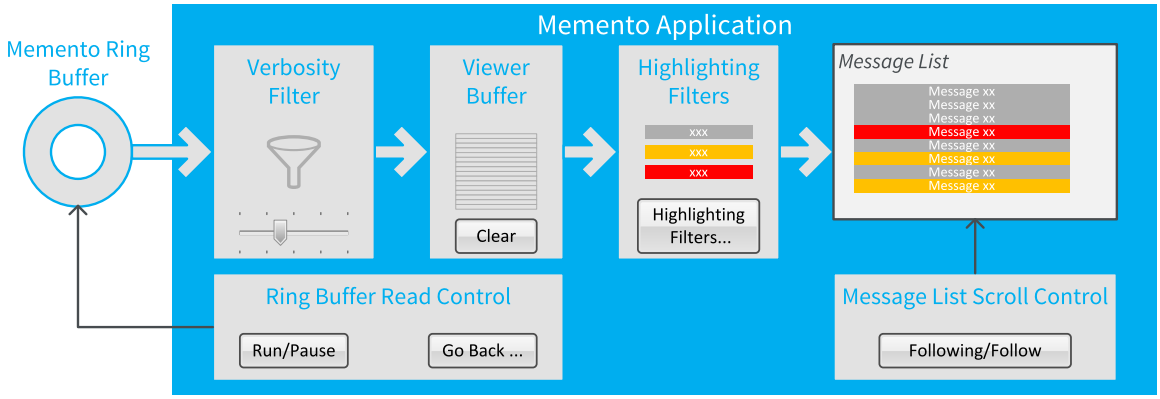
- Graphical User Interface
- Console Viewer

The Memento Viewer function allows to:

- View/change the verbosity level
- View/change the message filter configuration
- View the body text and the attributes of the selected messages
- Manage the Message Viewer memory
- Go back in Memento Ring Buffer history

8.1. Block Diagram

The following drawing shows the function blocks of the Memento Viewer.



Memento Application – VIEW Mode Function Blocks and Data Flow

The Memento Viewer processes the messages as follows:

- Reads messages from the Memento Ring
- Stores the messages satisfying the rules of the Verbosity Filter into the Viewer Buffer

- Reads messages from the Viewer Buffer
- Highlights the messages according to the settings of the Highlighting Filters
- Displays the highlighted messages in the **Message List Area** of the Memento GUI or on the console.

8.2. Buffer

The Viewer Buffer is a memory area where the Memento Viewer stores the messages to display.

The Viewer Buffer is cleared at start-up of the Memento Application. Then, it accumulates all the incoming Ring Buffer messages that satisfy the rules of the Verbosity Filter.

The Viewer Buffer has a capacity of 250,000 message slots. This can be modified with the command line argument "history", a higher value means a higher memory consumption.

The Viewer Buffer can be cleared at any time by the user using the "Clear" control button of Memento GUI.

Buffer Controls

By default, the Viewer Buffer is filled with the most recent messages of the Memento Ring Buffer.

However, other options are available using the **Run|Pause** and the **Go Back** controls.

Run/Pause Control

The **Run/Pause** control allows to pause the extraction of the messages from the Memento Ring Buffer and the filling process of the Viewer Buffer.

After a pause, the message extraction resumes.

> **Note:** *In the meantime messages can be lost if the ring capacity has been reached.*

Go Back Control

The **Go Back** control allows to reload the viewer buffer with older messages of the Memento Ring Buffer.

8.3. Verbosity Filter

The Verbosity Filter is inserted in the message path between the Memento Ring Buffer and the Viewer Buffer.

The Verbosity Filter allows to display only the messages that have a severity level (`level` attribute) greater than- or equal to- the verbosity level.

Verbosity Level	**Accepted Message Severity Levels**
Critical – *Lowest verbosity*	Critical – *Highest severity*
Error	Critical Error
Warning	Critical Error Warning
Notice	Critical Error Warning Notice
Info	Critical Error Warning Notice Info
Debug	Critical Error Warning Notice Info Debug
Verbose – *Highest verbosity*	*All levels accepted*

*The default verbosity level is **Info**.*

In Memento GUI, the Verbosity Filter is controlled by a slider. Moving the slider instantaneously adapts the filter rule and consequently the message flow.

8.4. Highlighting Filters

Sixteen Highlighting Filters can be inserted in the message path between the Viewer Buffer and the display output.

Each highlighting filter has two user-defined controls:

- A **filter rule** that selects a specific class of message
- A **filter action** that defines the highlighting action to be perform on the selected class of messages.

Filter Rule

The rule of an Highlighting Filter is defined using the following settings:

- An enable control
- A polarity control
- Six message attribute conditions
- One message content condition

Enable Control

The Enable Control allows to enable/disable each filter individually while keeping the other filter settings unchanged.

When Enable Control is unchecked, the highlighting filter is bypassed!

Polarity Control

The Polarity Control determines whether or not the action has to be executed when all the enabled conditions are True:

- When unchecked, the action is executed only when all the enabled conditions are True.
- When checked, the action is always executed except when all the enabled conditions are True.

Attribute Conditions

There is one condition for each of the following message attributes: `Level`, `Kind`, `PID`, `TID`, `Card`, `Connector`.

Each condition has an enable control and a value.

The condition is ignored when the enable control is unchecked.

The attribute condition is evaluated to TRUE when the corresponding message attribute value is equal to the condition value.

Message Content Condition

There is one message content condition. It is an enable control and a text string value.

The condition is ignored when the enable control is unchecked.

The message content condition is evaluated to TRUE when the message body contains the specified text.

Filter Action

The action of an Highlighting Filter is defined using the following settings:

- A hide control
- A background color selector
- A font color selector

Hide Control

The Hide Control determines whether or not messages of the selected class have to be hidden.

When the Hide Control is checked, the messages of the selected class are hidden.

When the Hide Control is unchecked, the messages of the selected class are displayed using the selected font and background colors.

The Hide Control is unchecked by default.

Background and Font Color Selectors

The selector proposes a set of basic and custom colors.

The default colors are:

- White background
- Black font

Default Settings

By default, the highlighting filters are configured as follows:

Highlighting Filters Default Configuration

Filter #	Filter Rule	Filter Action
1 to 10	*Disabled*	
10	Level=User	Highlight – Green text | white background
11	Level=Error	Highlight – Black text | red background
12	Level=Warning	Highlight – Black text | orange background
13	Level=Notice	Highlight – Green text | white background
14	Level=Info	Highlight – Purple text | white background
15	Level=Debug	Highlight – Grey text | white background
16	Level=Verbose	Highlight – Light Grey text | white background

8.5. Message List

The Memento Viewer composes the Message List with messages coming from the Viewer Buffer through the Highlighting Filters.

The messages are sorted by increasing time order if the time attribute is available otherwise by sequential order of arrival into the Memento Ring.

Messages being hidden by the Highlighting filter are not displayed in the message list.

Scroll Control

By default the message list is updated as soon as a new message arrives in the Viewer Buffer: the message list follows the Viewer Buffer. This mode of operation is named **auto-scroll**.

This auto-scroll mode can be disabled to allow older messages of the Message List to be read.

On Memento GUI, the **Follow|Following** toggle button enables or disables the **auto-scroll**.

9. Dump

The Memento Ring Buffer data can be dumped to disk files using the Memento Dump function of the Memento Application.

The Memento Viewer function is available in the following two modes of Memento Application:

- Graphical User Interface
- Console Dumper

Data Selection

By default, the Memento Dump function dumps the next-to-come Memento Ring data.

However, other options are available using the **Rewind** and the **Boot Session** options.

Dump Files Management

Dump File Name & Location

The Memento Dump function stores the Memento data into a disk file.

The default file name is `dump.memento` and the default file location is the Memento Application directory. `%USERPROFILE%\memento\`.

The file name and the file location are user configurable using the **File** option.

Euresys recommends to keep the suffix `.memento`.

Dump File Name Rotation

Memento Dump keeps previous dump files in the same directory by assigning a numerical suffix to their name. For instance, assuming the file name is set to `dump.memento`, older files are named `dump.1.memento`, `dump.2.memento`, `dump.3.memento`...

Every time a new dump file is created, the Memento Dump function

- Increments the numerical suffix of all existing dump files (file name rotation)
- Renames the previous dump file by adding the '.1' suffix

Dump File Size Control

By default, the Memento Dump function uses a single file per dump session.

Optionally, the user may split the dumped data over multiple files by specifying a size limit with the **Split size** option.

Dump File Count Control

By default, the Memento Dump function keeps all the dump files.

Optionally, the user may limit the number of dump files to keep using the **Keep** option.

Start/Stop Control

The dump operation starts on a user action:

- **Memento GUI:** by clicking the Start Button in the Dump dialog box.
- **Command Line:** by executing the Memento Application in the dump mode.

The dump operation stops:

- **Memento GUI:** by clicking the Stop Button in the Dump dialog box.
- **Memento GUI & Command Line**: on application exit.