



YAODAQ

Y e t A n O t h e r D A Q

Generated by Doxygen 1.9.3

1 License	1
2 Third-party licenses	3
2.1 LICENSE ISSUES	4
2.1.1 OpenSSL License	4
2.1.2 Original SSLeay License	6
3 Namespace Index	11
3.1 Namespace List	11
4 Hierarchical Index	13
4.1 Class Hierarchy	13
5 Data Structure Index	15
5.1 Data Structures	15
6 File Index	17
6.1 File List	17
7 Namespace Documentation	19
7.1 spdlog Namespace Reference	19
7.1.1 Detailed Description	19
7.1.2 Typedef Documentation	19
7.1.2.1 sink_ptr	19
7.2 yaodag Namespace Reference	19
7.2.1 Detailed Description	20
7.2.2 Enumeration Type Documentation	20
7.2.2.1 Class	20
7.2.2.2 Severity	20
7.2.2.3 Signal	20
7.2.2.4 StatusCode	21
8 Data Structure Documentation	23
8.1 yaodag::Exception Class Reference	23
8.1.1 Detailed Description	23
8.1.2 Constructor & Destructor Documentation	23
8.1.2.1 Exception() [1/2]	23
8.1.2.2 Exception() [2/2]	24
8.1.2.3 ~Exception()	24
8.1.3 Member Function Documentation	24
8.1.3.1 code()	24
8.1.3.2 description()	24
8.1.3.3 setFormat()	24
8.1.3.4 setStyle()	24
8.1.3.5 what()	24

8.2 yaodaq::Identifier Class Reference	24
8.2.1 Detailed Description	25
8.2.2 Constructor & Destructor Documentation	25
8.2.2.1 Identifier() [1/2]	25
8.2.2.2 Identifier() [2/2]	25
8.2.3 Member Function Documentation	25
8.2.3.1 get()	25
8.2.3.2 getClass()	25
8.2.3.3 getClassId()	25
8.2.3.4 getName()	26
8.2.3.5 getType()	26
8.3 yaodaq::Interrupt Class Reference	26
8.3.1 Detailed Description	26
8.3.2 Constructor & Destructor Documentation	26
8.3.2.1 Interrupt()	26
8.3.2.2 ~Interrupt()	26
8.3.3 Member Function Documentation	26
8.3.3.1 getSignal()	27
8.3.3.2 init()	27
8.3.3.3 restore()	27
8.4 yaodaq::LoggerHandler Class Reference	27
8.4.1 Detailed Description	28
8.4.2 Member Enumeration Documentation	28
8.4.2.1 Verbosity	28
8.4.3 Constructor & Destructor Documentation	28
8.4.3.1 LoggerHandler() [1/2]	28
8.4.3.2 LoggerHandler() [2/2]	28
8.4.3.3 ~LoggerHandler()	28
8.4.4 Member Function Documentation	28
8.4.4.1 addSink()	29
8.4.4.2 clearSinks()	29
8.4.4.3 logger()	29
8.4.4.4 setVerbosity()	29
8.5 yaodaq::Looper Class Reference	29
8.5.1 Detailed Description	29
8.5.2 Constructor & Destructor Documentation	30
8.5.2.1 Looper()	30
8.5.2.2 ~Looper()	30
8.5.3 Member Function Documentation	30
8.5.3.1 getInstance()	30
8.5.3.2 getSignal()	30
8.5.3.3 loop()	30

8.5.3.4 supressInstance()	30
8.6 yaodag::Version Class Reference	31
8.6.1 Detailed Description	31
8.6.2 Constructor & Destructor Documentation	31
8.6.2.1 Version() [1/3]	31
8.6.2.2 Version() [2/3]	31
8.6.2.3 Version() [3/3]	32
8.6.3 Member Function Documentation	32
8.6.3.1 getMajor()	32
8.6.3.2 getMinor()	32
8.6.3.3 getPatch()	32
8.6.3.4 getPreRelease()	32
8.6.3.5 getPreReleaseNumber()	32
8.7 yaodag::WebsocketClient Class Reference	32
8.7.1 Detailed Description	33
8.7.2 Constructor & Destructor Documentation	33
8.7.2.1 WebsocketClient()	33
8.7.2.2 ~WebsocketClient()	33
8.7.3 Member Function Documentation	33
8.7.3.1 logger()	33
8.7.3.2 loop()	33
8.7.3.3 start()	34
8.7.3.4 stop()	34
8.8 yaodag::WebsocketServer Class Reference	34
8.8.1 Detailed Description	34
8.8.2 Constructor & Destructor Documentation	35
8.8.2.1 WebsocketServer()	35
8.8.2.2 ~WebsocketServer()	35
8.8.3 Member Function Documentation	35
8.8.3.1 listen()	36
8.8.3.2 logger()	36
8.8.3.3 loop()	36
8.8.3.4 setVerbosity()	36
8.8.3.5 start()	36
8.8.3.6 stop()	36
9 File Documentation	39
9.1 docs/License.md File Reference	39
9.2 docs/Third-party licenses.md File Reference	39
9.3 yaodag/Class.hpp File Reference	39
9.4 Class.hpp	39
9.5 yaodag/Exception.hpp File Reference	40

9.6 Exception.hpp	40
9.7 yaodaq/Identifier.hpp File Reference	40
9.8 Identifier.hpp	41
9.9 yaodaq/Interrupt.hpp File Reference	41
9.10 Interrupt.hpp	41
9.11 yaodaq/LoggerHandler.hpp File Reference	42
9.12 LoggerHandler.hpp	42
9.13 yaodaq/Looper.hpp File Reference	43
9.14 Looper.hpp	43
9.15 yaodaq/Severity.hpp File Reference	44
9.16 Severity.hpp	44
9.17 yaodaq/Signal.hpp File Reference	44
9.18 Signal.hpp	44
9.19 yaodaq/StatusCode.hpp File Reference	45
9.20 StatusCode.hpp	45
9.21 yaodaq/Version.hpp File Reference	45
9.22 Version.hpp	46
9.23 yaodaq/WebsocketClient.hpp File Reference	46
9.24 WebsocketClient.hpp	46
9.25 yaodaq/WebsocketServer.hpp File Reference	47
9.26 WebsocketServer.hpp	47
9.27 yaodaq/Exception.cpp File Reference	48
9.28 Exception.cpp	48
9.29 yaodaq/Identifier.cpp File Reference	49
9.30 Identifier.cpp	49
9.31 yaodaq/Interrupt.cpp File Reference	49
9.32 Interrupt.cpp	49
9.33 yaodaq/LoggerHandler.cpp File Reference	50
9.34 LoggerHandler.cpp	51
9.35 yaodaq/Looper.cpp File Reference	51
9.36 Looper.cpp	51
9.37 yaodaq/Version.cpp File Reference	52
9.38 Version.cpp	52
9.39 yaodaq/WebsocketClient.cpp File Reference	53
9.40 WebsocketClient.cpp	53
9.41 yaodaq/WebsocketServer.cpp File Reference	54
9.42 WebsocketServer.cpp	54

Chapter 1

License

Copyright (c) 2022 YAODAO

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 2

Third-party licenses

The following software may be included in this product: CPMLicenses. This software contains the following license and notice below:

MIT License

Copyright (c) 2021 Lars Melchior

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: magic_enum. This software contains the following license and notice below:

MIT License

Copyright (c) 2019 - 2021 Daniil Goncharov

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: zlib-ng. This software contains the following license and notice below:

(C) 1995-2013 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

The following software may be included in this product: OpenSSL-CMake. This software contains the following license and notice below:

MIT License

Copyright (c) 2020 flagarde

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: OpenSSL. This software contains the following license and notice below:

2.1 LICENSE ISSUES

The OpenSSL toolkit stays under a double license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts.

2.1.1 OpenSSL License

/* =====

- Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved.
-
- Redistribution and use in source and binary forms, with or without
- modification, are permitted provided that the following conditions
- are met:
-
- 1. Redistributions of source code must retain the above copyright
- notice, this list of conditions and the following disclaimer.
-
- 2. Redistributions in binary form must reproduce the above copyright
- notice, this list of conditions and the following disclaimer in
- the documentation and/or other materials provided with the
- distribution.
-

- 3. All advertising materials mentioning features or use of this
- software must display the following acknowledgment:
- "This product includes software developed by the OpenSSL Project * for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)"
-
- 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
- endorse or promote products derived from this software without
- prior written permission. For written permission, please contact
- openssl-core@openssl.org.
-
- 5. Products derived from this software may not be called "OpenSSL"
- nor may "OpenSSL" appear in their names without prior written
- permission of the OpenSSL Project.
-
- 6. Redistributions of any form whatsoever must retain the following
- acknowledgment:
- "This product includes software developed by the OpenSSL Project * for use in the OpenSSL Toolkit (<http://www.openssl.org/>)"
-
- THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
- EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
- PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
- ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
- SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
- NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
- LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
- HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
- STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
- ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
- OF THE POSSIBILITY OF SUCH DAMAGE.
- =====
-
- This product includes cryptographic software written by Eric Young
- (eay@cryptsoft.com). This product includes software written by Tim
- Hudson (tjh@cryptsoft.com).
- */

2.1.2 Original SSLeay License

/* Copyright (C) 1995-1998 Eric Young (eyay@cryptsoft.com)

- All rights reserved.
-
- This package is an SSL implementation written
- by Eric Young (eyay@cryptsoft.com).
- The implementation was written so as to conform with Netscapes SSL.
-
- This library is free for commercial and non-commercial use as long as
- the following conditions are aheared to. The following conditions
- apply to all code found in this distribution, be it the RC4, RSA,
- lhash, DES, etc., code; not just the SSL code. The SSL documentation
- included with this distribution is covered by the same copyright terms
- except that the holder is Tim Hudson (tjh@cryptsoft.com).
-
- Copyright remains Eric Young's, and as such any Copyright notices in
- the code are not to be removed.
- If this package is used in a product, Eric Young should be given attribution
- as the author of the parts of the library used.
- This can be in the form of a textual message at program startup or
- in documentation (online or textual) provided with the package.
-
- Redistribution and use in source and binary forms, with or without
- modification, are permitted provided that the following conditions
- are met:
- 1. Redistributions of source code must retain the copyright
- notice, this list of conditions and the following disclaimer.
- 2. Redistributions in binary form must reproduce the above copyright
- notice, this list of conditions and the following disclaimer in the
- documentation and/or other materials provided with the distribution.
- 3. All advertising materials mentioning features or use of this software
- must display the following acknowledgement:
- "This product includes cryptographic software written by * Eric Young (eyay@cryptsoft.com)"
- The word 'cryptographic' can be left out if the rouines from the library
- being used are not cryptographic related :-).
- 4. If you include any Windows specific code (or a derivative thereof) from

- the apps directory (application code) you must include an acknowledgement:
- "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
-
- THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS" AND
- ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
- IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
- ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
- FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
- DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
- OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
- HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
- LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
- OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
- SUCH DAMAGE.
-
- The licence and distribution terms for any publically available version or
- derivative of this code cannot be changed. i.e. this code cannot simply be
- copied and put under another distribution licence
- [including the GNU Public Licence.] */

The following software may be included in this product: IXWebSocket. This software contains the following license and notice below:

Copyright (c) 2018 Machine Zone, Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following software may be included in this product: fmt. This software contains the following license and notice below:

Copyright (c) 2012 - present, Victor Zverovich

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights

to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

— Optional exception to the license —

As an exception, if, as a result of your compiling your source code, portions of this Software are embedded into a machine-executable object form of such source code, you may redistribute such embedded portions in such object form without including the above copyright and permission notices.

The following software may be included in this product: spdlog. This software contains the following license and notice below:

The MIT License (MIT)

Copyright (c) 2016 Gabi Melman.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

– NOTE: Third party dependency used by this software – This software depends on the fmt lib (MIT License), and users must comply to its license: <https://github.com/fmtlib/fmt/blob/master/LICENSE.rst>

The following software may be included in this product: nlohmann. This software contains the following license and notice below:

MIT License

Copyright (c) 2013-2022 Niels Lohmann

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: SourceLocation. This software contains the following license and notice below:

MIT License

Copyright (c) 2021 flagarde

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the

Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: Semver. This software contains the following license and notice below:

MIT License

Copyright (c) 2018 - 2021 Daniil Goncharov

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following software may be included in this product: CLI11. This software contains the following license and notice below:

CLI11 1.8 Copyright (c) 2017-2019 University of Cincinnati, developed by Henry Schreiner under NSF AWARD 1414736. All rights reserved.

Redistribution and use in source and binary forms of CLI11, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following software may be included in this product: doctest. This software contains the following license and notice below:

The MIT License (MIT)

Copyright (c) 2016-2021 Viktor Kirilov

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE

LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

spdlog	19
yaodaq	19

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

std::exception	
yaodag::Exception	23
yaodag::Identifier	24
yaodag::Interrupt	26
yaodag::LoggerHandler	27
yaodag::Looper	29
source_location	
yaodag::Exception	23
semver::version	
yaodag::Version	31
ix::WebSocket	
yaodag::WebsocketClient	32
ix::WebSocketServer	
yaodag::WebsocketServer	34

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

yaodag::Exception	23
yaodag::Identifier	24
yaodag::Interrupt	26
yaodag::LoggerHandler	27
yaodag::Looper	29
yaodag::Version	31
yaodag::WebsocketClient	32
yaodag::WebsocketServer	34

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

yaodag/Class.hpp	39
yaodag/Exception.hpp	40
yaodag/Identifier.hpp	40
yaodag/Interrupt.hpp	41
yaodag/LoggerHandler.hpp	42
yaodag/Looper.hpp	43
yaodag/Severity.hpp	44
yaodag/Signal.hpp	44
yaodag/StatusCode.hpp	45
yaodag/Version.hpp	45
yaodag/WebsocketClient.hpp	46
yaodag/WebsocketServer.hpp	47
yaodag/Exception.cpp	48
yaodag/Identifier.cpp	49
yaodag/Interrupt.cpp	49
yaodag/LoggerHandler.cpp	50
yaodag/Looper.cpp	51
yaodag/Version.cpp	52
yaodag/WebsocketClient.cpp	53
yaodag/WebsocketServer.cpp	54

Chapter 7

Namespace Documentation

7.1 spdlog Namespace Reference

Typedefs

- using [sink_ptr](#) = std::shared_ptr< spdlog::sinks::sink >

7.1.1 Detailed Description

Copyright

Copyright 2022 flagarde

7.1.2 Typedef Documentation

7.1.2.1 sink_ptr

using [spdlog::sink_ptr](#) = typedef std::shared_ptr<spdlog::sinks::sink>
Definition at line 15 of file [LoggerHandler.hpp](#).

7.2 yaodaq Namespace Reference

Data Structures

- class [Exception](#)
- class [Identifier](#)
- class [Interrupt](#)
- class [LoggerHandler](#)
- class [Looper](#)
- class [Version](#)
- class [WebsocketClient](#)
- class [WebsocketServer](#)

Enumerations

- enum class [Class](#) : std::int_least16_t {
 [Unknown](#) = -1 , [Module](#) = 0 , [Browser](#) = 100 , [WebsocketServer](#) = Module + 1 ,
 [WebsocketClient](#) = Module + 2 }
- enum class [Severity](#) : std::int_least16_t { [Info](#) = 1 , [Warning](#) = 10 , [Error](#) = 100 , [Critical](#) = 1000 }
- enum class [Signal](#) {
 [NO](#) = 0 , [ABRT](#) = static_cast<int>(Severity::Critical) + 1 , [FPE](#) = static_cast<int>(Severity::Critical) + 2 ,
 [ILL](#) = static_cast<int>(Severity::Critical) + 3 ,

```
SEGV = static_cast<int>( Severity::Critical ) + 4 , INT = static_cast<int>( Severity::Warning ) + 1 , TERM =
static_cast<int>( Severity::Warning ) + 2 }
```

- enum class [StatusCode](#) : std::int_least32_t { [SUCCESS](#) = 0 , [LISTEN_ERROR](#) }

7.2.1 Detailed Description

Copyright

Copyright 2022 flagarde

7.2.2 Enumeration Type Documentation

7.2.2.1 Class

```
enum class yaodag::Class : std::int_least16_t [strong]
```

Enumerator

Unknown	
Module	
Browser	
WebsocketServer	
WebsocketClient	

Definition at line 13 of file [Class.hpp](#).

```
00014 {
00015     Unknown = -1,
00016     Module  = 0,
00017     Browser  = 100,
00018
00019     WebsocketServer = Module + 1,
00020     WebsocketClient = Module + 2,
00021 };
```

7.2.2.2 Severity

```
enum class yaodag::Severity : std::int_least16_t [strong]
```

Enumerator

Info	
Warning	
Error	
Critical	

Definition at line 13 of file [Severity.hpp](#).

```
00014 {
00015     Info    = 1,
00016     Warning = 10,
00017     Error   = 100,
00018     Critical = 1000,
00019 };
```

7.2.2.3 Signal

```
enum class yaodag::Signal [strong]
```

Enumerator

NO	
ABRT	
FPE	
ILL	
SEGV	
INT	
TERM	

Definition at line 15 of file [Signal.hpp](#).

```

00016 {
00017     NO    = 0,    // No Signal.
00018     // Critical
00019     ABRT = static_cast<int>( Severity::Critical ) + 1,    // (Signal Abort) Abnormal termination, such as
                is initiated by the abort function.
00020     FPE  = static_cast<int>( Severity::Critical ) + 2,    // (Signal Floating-Point Exception) Erroneous
                arithmetic operation, such as zero divide or an operation resulting in overflow (not necessarily with
                a floating-point operation).
00021     ILL  = static_cast<int>( Severity::Critical ) + 3,    // (Signal Illegal Instruction) Invalid function
                image, such as an illegal instruction. This is generally due to a corruption in the code or to an
                attempt to execute data.
00022     SEGV = static_cast<int>( Severity::Critical ) + 4,    // (Signal Segmentation Violation) Invalid
                access to storage: When a program tries to read or write outside the memory it has allocated.
00023     // Warning
00024     INT  = static_cast<int>( Severity::Warning ) + 1,    // (Signal Interrupt) Interactive attention
                signal. Generally generated by the application user.
00025     TERM = static_cast<int>( Severity::Warning ) + 2,    // (Signal Terminate) Termination request sent to
                program.
00026 };

```

7.2.2.4 StatusCode

```
enum class yaodag::StatusCode : std::int_least32_t    [strong]
```

Enumerator

SUCCESS	
LISTEN_ERROR	

Definition at line 13 of file [StatusCode.hpp](#).

```

00014 {
00015     SUCCESS = 0,
00016     LISTEN_ERROR,
00017 };

```

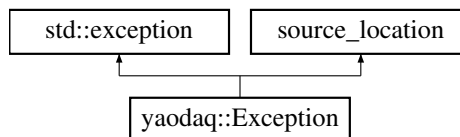

Chapter 8

Data Structure Documentation

8.1 yaodaq::Exception Class Reference

```
#include <yaodaq/Exception.hpp>
```

Inheritance diagram for yaodaq::Exception:



Public Member Functions

- [Exception](#) ()=delete
- [Exception](#) (const [StatusCode](#) &statusCode, const std::string &[description](#), const source_location &location=source_location::current())
- [~Exception](#) () noexcept override=default
- const char * [what](#) () const noexcept final
- const char * [description](#) () const noexcept
- std::int_least32_t [code](#) () const noexcept

Static Public Member Functions

- static void [setFormat](#) (const std::string &format)
- static void [setStyle](#) (const fmt::text_style &style={})

8.1.1 Detailed Description

Definition at line 19 of file [Exception.hpp](#).

8.1.2 Constructor & Destructor Documentation

8.1.2.1 Exception() [1/2]

```
yaodaq::Exception::Exception ( ) [delete]
```

8.1.2.2 Exception() [2/2]

```
yaodaq::Exception::Exception (
    const StatusCode & statusCode,
    const std::string & description,
    const source_location & location = source_location::current() )
```

Definition at line 14 of file [Exception.cpp](#).

```
00014 : source_location( location ), m_Code( static\_cast<std::int_least32_t>( statusCode ) ), m_Description(
    description ) { constructMessage(); }
```

8.1.2.3 ~Exception()

```
yaodaq::Exception::~~Exception ( ) [override], [default], [noexcept]
```

8.1.3 Member Function Documentation

8.1.3.1 code()

```
int_least32_t yaodaq::Exception::code ( ) const [noexcept]
```

Definition at line 20 of file [Exception.cpp](#).

```
00020 { return m_Code; }
```

8.1.3.2 description()

```
const char * yaodaq::Exception::description ( ) const [noexcept]
```

Definition at line 18 of file [Exception.cpp](#).

```
00018 { return m_Description.c_str(); }
```

8.1.3.3 setFormat()

```
static void yaodaq::Exception::setFormat (
    const std::string & format ) [inline], [static]
```

Definition at line 24 of file [Exception.hpp](#).

```
00024 { m_Format = format; }
```

8.1.3.4 setStyle()

```
static void yaodaq::Exception::setStyle (
    const fmt::text_style & style = {} ) [inline], [static]
```

Definition at line 26 of file [Exception.hpp](#).

```
00026 {} ) { m_Style = style; }
```

8.1.3.5 what()

```
const char * yaodaq::Exception::what ( ) const [final], [noexcept]
```

Definition at line 16 of file [Exception.cpp](#).

```
00016 { return m_Message.c_str(); }
```

The documentation for this class was generated from the following files:

- [yaodaq/Exception.hpp](#)
- [yaodaq/Exception.cpp](#)

8.2 yaodaq::Identifier Class Reference

```
#include <yaodaq/Identifier.hpp>
```

Public Member Functions

- [Identifier](#) ()=default
- [Identifier](#) (const [Class](#) &aClass, const std::string &type, const std::string &name)
- std::string [getClass](#) () const
- std::string [getType](#) () const
- std::string [getName](#) () const
- [Class](#) [getClassId](#) () const
- std::string [get](#) () const

8.2.1 Detailed Description

Definition at line 15 of file [Identifier.hpp](#).

8.2.2 Constructor & Destructor Documentation

8.2.2.1 Identifier() [1/2]

```
yaodaq::Identifier::Identifier ( ) [default]
```

8.2.2.2 Identifier() [2/2]

```
yaodaq::Identifier::Identifier (
    const Class & aClass,
    const std::string & type,
    const std::string & name )
```

Definition at line 16 of file [Identifier.cpp](#).

```
00016 : m_Class( aClass ), m_Type( type ), m_Name( name ) {}
```

8.2.3 Member Function Documentation

8.2.3.1 get()

```
std::string yaodaq::Identifier::get ( ) const
```

Definition at line 26 of file [Identifier.cpp](#).

```
00026 { return fmt::format( "{0}/{1}/{2}", getClass(), getType(), getName() ); }
```

8.2.3.2 getClass()

```
std::string yaodaq::Identifier::getClass ( ) const
```

Definition at line 18 of file [Identifier.cpp](#).

```
00018 { return std::string( magic_enum::enum_name( m_Class ) ); }
```

8.2.3.3 getClassId()

```
Class yaodaq::Identifier::getClassId ( ) const
```

Definition at line 24 of file [Identifier.cpp](#).

```
00024 { return m_Class; }
```

8.2.3.4 getName()

```
std::string yaodaq::Identifier::getName ( ) const
```

Definition at line 22 of file [Identifier.cpp](#).

```
00022 { return m_Name; }
```

8.2.3.5 getType()

```
std::string yaodaq::Identifier::getType ( ) const
```

Definition at line 20 of file [Identifier.cpp](#).

```
00020 { return m_Type; }
```

The documentation for this class was generated from the following files:

- [yaodaq/Identifier.hpp](#)
- [yaodaq/Identifier.cpp](#)

8.3 yaodaq::Interrupt Class Reference

```
#include <yaodaq/Interrupt.hpp>
```

Public Member Functions

- [Interrupt \(\)](#)
- void [init \(\)](#)
- void [restore \(\)](#)
- [Signal getSignal \(\)](#)
- [~Interrupt \(\)](#)

8.3.1 Detailed Description

Definition at line 19 of file [Interrupt.hpp](#).

8.3.2 Constructor & Destructor Documentation

8.3.2.1 Interrupt()

```
yaodaq::Interrupt::Interrupt ( )
```

Definition at line 19 of file [Interrupt.cpp](#).

```
00019 { init(); }
```

8.3.2.2 ~Interrupt()

```
yaodaq::Interrupt::~~Interrupt ( )
```

Definition at line 42 of file [Interrupt.cpp](#).

```
00042 { restore(); }
```

8.3.3 Member Function Documentation

8.3.3.1 getSignal()

`Signal yaodaq::Interrupt::getSignal ()`

Definition at line 44 of file [Interrupt.cpp](#).

```

00045 {
00046     if( m_Signal.load() != Signal::NO )
00047     {
00048         std::lock_guard<std::mutex> guard( m_mutex );
00049         init();
00050     }
00051     return m_Signal.load();
00052 }
```

8.3.3.2 init()

`void yaodaq::Interrupt::init ()`

Definition at line 31 of file [Interrupt.cpp](#).

```

00032 {
00033     setSignal( Signal::TERM );
00034     setSignal( Signal::TERM );
00035     setSignal( Signal::SEGV );
00036     setSignal( Signal::INT );
00037     setSignal( Signal::ILL );
00038     setSignal( Signal::ABRT );
00039     setSignal( Signal::FPE );
00040 }
```

8.3.3.3 restore()

`void yaodaq::Interrupt::restore ()`

Definition at line 21 of file [Interrupt.cpp](#).

```

00022 {
00023     std::signal( SIGTERM, SIG_DFL );
00024     std::signal( SIGSEGV, SIG_DFL );
00025     std::signal( SIGINT, SIG_DFL );
00026     std::signal( SIGILL, SIG_DFL );
00027     std::signal( SIGABRT, SIG_DFL );
00028     std::signal( SIGFPE, SIG_DFL );
00029 }
```

The documentation for this class was generated from the following files:

- [yaodaq/Interrupt.hpp](#)
- [yaodaq/Interrupt.cpp](#)

8.4 yaodaq::LoggerHandler Class Reference

```
#include <yaodaq/LoggerHandler.hpp>
```

Public Types

- enum class [Verbosity](#) {
 [Off](#) , [Trace](#) , [Debug](#) , [Info](#) ,
 [Warn](#) , [Error](#) , [Critical](#) }

Public Member Functions

- [LoggerHandler](#) ()
- [LoggerHandler](#) (const std::string &)
- [~LoggerHandler](#) ()
- void [setVerbosity](#) (const [Verbosity](#) &verbosity)
- std::shared_ptr< spdlog::logger > [logger](#) ()
- void [addSink](#) (const [spdlog::sink_ptr](#) &)
- void [clearSinks](#) ()

8.4.1 Detailed Description

Definition at line 21 of file [LoggerHandler.hpp](#).

8.4.2 Member Enumeration Documentation

8.4.2.1 Verbosity

```
enum class yaodag::LoggerHandler::Verbosity [strong]
```

Enumerator

Off	
Trace	
Debug	
Info	
Warn	
Error	
Critical	

Definition at line 24 of file [LoggerHandler.hpp](#).

```
00025 {
00026     Off,
00027     Trace,
00028     Debug,
00029     Info,
00030     Warn,
00031     Error,
00032     Critical
00033 };
```

8.4.3 Constructor & Destructor Documentation

8.4.3.1 LoggerHandler() [1/2]

```
yaodag::LoggerHandler::LoggerHandler ( )
```

Definition at line 12 of file [LoggerHandler.cpp](#).

```
00012 { init(); }
```

8.4.3.2 LoggerHandler() [2/2]

```
yaodag::LoggerHandler::LoggerHandler (
    const std::string & name ) [explicit]
```

Definition at line 14 of file [LoggerHandler.cpp](#).

```
00014 : m_Name( name ) { init(); }
```

8.4.3.3 ~LoggerHandler()

```
yaodag::LoggerHandler::~~LoggerHandler ( )
```

Definition at line 16 of file [LoggerHandler.cpp](#).

```
00016 {}
```

8.4.4 Member Function Documentation

8.4.4.1 addSink()

```
void yaodag::LoggerHandler::addSink (
    const spdlog::sink\_ptr & sink )
```

Definition at line 41 of file [LoggerHandler.cpp](#).

```
00042 {
00043     m_Sinks.push_back( sink );
00044     init();
00045 }
```

8.4.4.2 clearSinks()

```
void yaodag::LoggerHandler::clearSinks ( )
```

Definition at line 47 of file [LoggerHandler.cpp](#).

```
00048 {
00049     m_Sinks.clear();
00050     init();
00051 }
```

8.4.4.3 logger()

```
std::shared_ptr< spdlog::logger > yaodag::LoggerHandler::logger ( )
```

Definition at line 39 of file [LoggerHandler.cpp](#).

```
00039 { return std::shared_ptr<spdlog::logger>( m_Logger ); }
```

8.4.4.4 setVerbosity()

```
void yaodag::LoggerHandler::setVerbosity (
    const Verbosity & verbosity )
```

Definition at line 18 of file [LoggerHandler.cpp](#).

```
00019 {
00020     m_Verbosity = verbosity;
00021     init();
00022 }
```

The documentation for this class was generated from the following files:

- [yaodag/LoggerHandler.hpp](#)
- [yaodag/LoggerHandler.cpp](#)

8.5 yaodag::Looper Class Reference

```
#include <yaodag/Looper.hpp>
```

Public Member Functions

- [Looper](#) ()
- [Signal loop](#) ()
- [Signal getSignal](#) ()
- void [supressInstance](#) ()
- [~Looper](#) ()

Static Public Member Functions

- static int [getInstance](#) ()

8.5.1 Detailed Description

Definition at line 15 of file [Looper.hpp](#).

8.5.2 Constructor & Destructor Documentation

8.5.2.1 Looper()

yaodaq::Looper::Looper ()

Definition at line 28 of file [Looper.cpp](#).

```
00029 {
00030     if( m_hasBeenAdded == false )
00031     {
00032         m_hasBeenAdded = true;
00033         ++m_instance;
00034     }
00035 }
```

8.5.2.2 ~Looper()

yaodaq::Looper::~~Looper ()

Definition at line 52 of file [Looper.cpp](#).

```
00053 {
00054     if( m_hasBeenAdded == true && m_hasBeenSupressed == false )
00055     {
00056         m_hasBeenSupressed = true;
00057         --m_instance;
00058     }
00059 }
```

8.5.3 Member Function Documentation

8.5.3.1 getInstance()

int yaodaq::Looper::getInstance () [static]

Definition at line 17 of file [Looper.cpp](#).

```
00017 { return m_instance; }
```

8.5.3.2 getSignal()

Signal yaodaq::Looper::getSignal ()

Definition at line 50 of file [Looper.cpp](#).

```
00050 { return m_interrupt.getSignal(); }
```

8.5.3.3 loop()

Signal yaodaq::Looper::loop ()

Definition at line 37 of file [Looper.cpp](#).

```
00038 {
00039     static Signal signal{ yaodaq::Signal::NO };
00040     if( m_instance == 0 )
00041     {
00042         do {
00043             signal = m_interrupt.getSignal();
00044             std::this_thread::sleep_for( std::chrono::microseconds( 1 ) );
00045         } while( signal == yaodaq::Signal::NO );
00046     }
00047     return signal;
00048 }
```

8.5.3.4 supressInstance()

void yaodaq::Looper::supressInstance ()

Definition at line 19 of file [Looper.cpp](#).

```
00020 {
```

```

00021     if( m_hasBeenSupressed == false )
00022     {
00023         m_hasBeenSupressed = true;
00024         m_instance--;
00025     }
00026 }

```

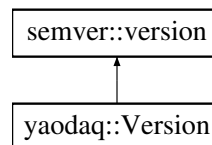
The documentation for this class was generated from the following files:

- [yaodag/Looper.hpp](#)
- [yaodag/Looper.cpp](#)

8.6 yaodag::Version Class Reference

```
#include <yaodag/Version.hpp>
```

Inheritance diagram for yaodag::Version:



Public Member Functions

- constexpr [Version](#) (const std::uint8_t &mj, const std::uint8_t &mn, const std::uint8_t &pt, const semver::prerelease &prr=semver::prerelease::none, const std::uint8_t &prn=0) noexcept
- constexpr [Version](#) (const std::string_view &str)
- constexpr [Version](#) ()=default
- std::uint8_t [getMajor](#) ()
- std::uint8_t [getMinor](#) ()
- std::uint8_t [getPatch](#) ()
- std::string [getPreRelease](#) ()
- std::uint8_t [getPreReleaseNumber](#) ()

8.6.1 Detailed Description

Definition at line 15 of file [Version.hpp](#).

8.6.2 Constructor & Destructor Documentation

8.6.2.1 Version() [1/3]

```

constexpr yaodag::Version::Version (
    const std::uint8_t & mj,
    const std::uint8_t & mn,
    const std::uint8_t & pt,
    const semver::prerelease & prt = semver::prerelease::none,
    const std::uint8_t & prn = 0 ) [inline], [constexpr], [noexcept]

```

Definition at line 18 of file [Version.hpp](#).

```
00018 : semver::version( mj, mn, pt, prt, prn ) {}
```

8.6.2.2 Version() [2/3]

```

constexpr yaodag::Version::Version (
    const std::string_view & str ) [inline], [explicit], [constexpr]

```

Definition at line 19 of file [Version.hpp](#).

```
00019 : semver::version( str ) {}
```

8.6.2.3 Version() [3/3]

```
constexpr yaodaq::Version::Version ( ) [constexpr], [default]
```

8.6.3 Member Function Documentation

8.6.3.1 getMajor()

```
std::uint8_t yaodaq::Version::getMajor ( )
```

Definition at line 12 of file [Version.cpp](#).

```
00012 { return major; }
```

8.6.3.2 getMinor()

```
std::uint8_t yaodaq::Version::getMinor ( )
```

Definition at line 14 of file [Version.cpp](#).

```
00014 { return minor; }
```

8.6.3.3 getPatch()

```
std::uint8_t yaodaq::Version::getPatch ( )
```

Definition at line 16 of file [Version.cpp](#).

```
00016 { return patch; }
```

8.6.3.4 getPreRelease()

```
std::string yaodaq::Version::getPreRelease ( )
```

Definition at line 18 of file [Version.cpp](#).

```
00018 { return std::string( magic_enum::enum_name( prerelease_type ) ); }
```

8.6.3.5 getPreReleaseNumber()

```
std::uint8_t yaodaq::Version::getPreReleaseNumber ( )
```

Definition at line 20 of file [Version.cpp](#).

```
00020 { return prerelease_number; }
```

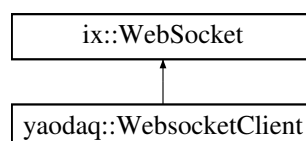
The documentation for this class was generated from the following files:

- [yaodaq/Version.hpp](#)
- [yaodaq/Version.cpp](#)

8.7 yaodaq::WebsocketClient Class Reference

```
#include <yaodaq/WebsocketClient.hpp>
```

Inheritance diagram for yaodaq::WebsocketClient:



Public Member Functions

- [WebsocketClient](#) (const std::string &name, const std::string &type="DefaultWebsocketClient")
- virtual [~WebsocketClient](#) ()
- void [start](#) ()
- void [stop](#) ()
- void [loop](#) ()
- std::shared_ptr< spdlog::logger > [logger](#) ()

8.7.1 Detailed Description

Definition at line 22 of file [WebsocketClient.hpp](#).

8.7.2 Constructor & Destructor Documentation

8.7.2.1 WebsocketClient()

```
yaodag::WebsocketClient::WebsocketClient (
    const std::string & name,
    const std::string & type = "DefaultWebsocketClient" ) [explicit]
```

Definition at line 16 of file [WebsocketClient.cpp](#).

```
00016         : m_Identifier(
00017             Class::WebsocketClient, type, name ), m_Logger( m_Identifier.get() )
00018 {
00019     ix::initNetSystem();
00020     ix::WebSocketHttpHeaders header{ { "Id", m_Identifier.get() } };
00021     setExtraHeaders( header );
00022     m_Logger.addSink( std::make_shared<spdlog::sinks::stdout_color_sink_mt>() );
00023     setOnMessageCallback(
00024         [this]( const ix::WebSocketMessagePtr& msg )
00025     {
00026         if( msg->type == ix::WebSocketMessageType::Message ) { logger()->error( "{}", msg->str ); }
00027     } );
```

8.7.2.2 ~WebsocketClient()

```
yaodag::WebsocketClient::~~WebsocketClient ( ) [virtual]
```

Definition at line 29 of file [WebsocketClient.cpp](#).

```
00030 {
00031     stop();
00032     ix::uninitNetSystem();
00033 }
```

8.7.3 Member Function Documentation

8.7.3.1 logger()

```
std::shared_ptr< spdlog::logger > yaodag::WebsocketClient::logger ( ) [inline]
```

Definition at line 30 of file [WebsocketClient.hpp](#).

```
00030 { return m_Logger.logger(); }
```

8.7.3.2 loop()

```
void yaodag::WebsocketClient::loop ( )
```

Definition at line 54 of file [WebsocketClient.cpp](#).

```
00055 {
00056     WebsocketClient::start();
00057     m_Looper.supressInstance();
00058     onRaisingSignal();
00059 }
```

8.7.3.3 start()

void yaodag::WebsocketClient::start ()

Definition at line 35 of file [WebsocketClient.cpp](#).

```
00036 {
00037     if( getReadyState() == ix::ReadyState::Closed || getReadyState() == ix::ReadyState::Closing )
00038     {
00039         logger()->trace( "Client started. Connected to {}", getUrl() );
00040         ix::WebSocket::start();
00041     }
00042 }
```

8.7.3.4 stop()

void yaodag::WebsocketClient::stop ()

Definition at line 44 of file [WebsocketClient.cpp](#).

```
00045 {
00046     if( getReadyState() == ix::ReadyState::Open || getReadyState() == ix::ReadyState::Connecting )
00047     {
00048         logger()->trace( "Client stopped" );
00049         ix::WebSocket::stop();
00050         while( getReadyState() != ix::ReadyState::Closed ) { std::this_thread::sleep_for(
std::chrono::microseconds( 1 ) ); }
00051     }
00052 }
```

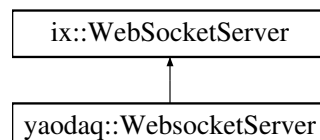
The documentation for this class was generated from the following files:

- [yaodag/WebsocketClient.hpp](#)
- [yaodag/WebsocketClient.cpp](#)

8.8 yaodag::WebsocketServer Class Reference

#include <yaodag/WebsocketServer.hpp>

Inheritance diagram for yaodag::WebsocketServer:



Public Member Functions

- [WebsocketServer](#) (const std::string &name, const int &port=ix::SocketServer::kDefaultPort, const std::string &host=ix::SocketServer::kDefaultHost, const int &backlog=ix::SocketServer::kDefaultTcpBacklog, const std::size_t &maxConnections=ix::SocketServer::kDefaultMaxConnections, const int &handshakeTimeoutSecs=ix::WebsocketServer::kDefaultHandShakeTimeoutSecs, const int &addressFamily=ix::SocketServer::kDefaultAddressFamily, const std::string &type="DefaultWebsocketServer")
- virtual [~WebsocketServer](#) ()
- void [loop](#) ()
- void [start](#) ()
- void [stop](#) (bool useless=true)
- void [listen](#) ()
- void [setVerbosity](#) (const yaodag::LoggerHandler::Verbosity &verbosity)
- std::shared_ptr< spdlog::logger > [logger](#) ()

8.8.1 Detailed Description

Definition at line 22 of file [WebsocketServer.hpp](#).

8.8.2 Constructor & Destructor Documentation

8.8.2.1 WebsocketServer()

```
yaodaq::WebsocketServer::WebsocketServer (
    const std::string & name,
    const int & port = ix::SocketServer::kDefaultPort,
    const std::string & host = ix::SocketServer::kDefaultHost,
    const int & backlog = ix::SocketServer::kDefaultTcpBacklog,
    const std::size_t & maxConnections = ix::SocketServer::kDefaultMaxConnections,
    const int & handshakeTimeoutSecs = ix::WebsocketServer::kDefaultHandShakeTimeoutSecs,
    const int & addressFamily = ix::SocketServer::kDefaultAddressFamily,
    const std::string & type = "DefaultWebsocketServer" ) [explicit]
```

Definition at line 22 of file [WebsocketServer.cpp](#).

```
00022
:
00023 ix::WebsocketServer( port, host, backlog, maxConnections, handshakeTimeoutSecs, addressFamily ),
m_Identifier( Class::WebsocketServer, type, name ), m_Logger( m_Identifier.get() )
00024 {
00025     ix::initNetSystem();
00026     m_Logger.addSink( std::make_shared<spdlog::sinks::stdout_color_sink_mt>() );
00027     setOnClientMessageCallback(
00028         []( std::shared_ptr<ix::ConnectionState> connectionState, ix::Websocket& websocket, const
ix::WebsocketMessagePtr& msg )
00029         {
00030             // The ConnectionState object contains information about the connection,
00031             // at this point only the client ip address and the port.
00032             std::cout << "Remote ip: " << connectionState->getRemoteIp() << std::endl;
00033
00034             if( msg->type == ix::WebsocketMessageType::Open )
00035             {
00036                 std::cout << "New connection" << std::endl;
00037
00038                 // A connection state object is available, and has a default id
00039                 // You can subclass ConnectionState and pass an alternate factory
00040                 // to override it. It is useful if you want to store custom
00041                 // attributes per connection (authenticated bool flag, attributes, etc...)
00042                 std::cout << "id: " << connectionState->getId() << std::endl;
00043
00044                 // The uri the client did connect to.
00045                 std::cout << "Uri: " << msg->openInfo.uri << std::endl;
00046
00047                 std::cout << "Headers:" << std::endl;
00048                 for( auto it: msg->openInfo.headers ) { std::cout << "\t" << it.first << ": " << it.second <<
std::endl; }
00049             }
00050             else if( msg->type == ix::WebsocketMessageType::Message )
00051             {
00052                 // For an echo server, we just send back to the client whatever was received by the server
00053                 // All connected clients are available in an std::set. See the broadcast cpp example.
00054                 // Second parameter tells whether we are sending the message in binary or text mode.
00055                 // Here we send it in the same mode as it was received.
00056                 std::cout << "Received: " << msg->str << std::endl;
00057
00058                 websocket.send( msg->str, msg->binary );
00059             }
00060         } );
00061 }
```

8.8.2.2 ~WebsocketServer()

```
yaodaq::WebsocketServer::~~WebsocketServer ( ) [virtual]
```

Definition at line 101 of file [WebsocketServer.cpp](#).

```
00102 {
00103     stop();
00104     ix::uninitNetSystem();
00105 }
```

8.8.3 Member Function Documentation

8.8.3.1 listen()

void yaodaq::WebsocketServer::listen ()

Definition at line 63 of file [WebsocketServer.cpp](#).

```
00064 {
00065     if( !m_isListening )
00066     {
00067         std::pair<bool, std::string> ret = ix::WebSocketServer::listen();
00068         if( ret.first )
00069         {
00070             m_isListening = ret.first;
00071             logger()->info( "Server listening on host {0} port {1}", getHost(), getPort() );
00072         }
00073         else
00074             throw Exception( StatusCode::LISTEN_ERROR, ret.second );
00075     }
00076 }
```

8.8.3.2 logger()

std::shared_ptr< spdlog::logger > yaodaq::WebsocketServer::logger () [inline]

Definition at line 36 of file [WebsocketServer.hpp](#).

```
00036 { return m_Logger.logger(); }
```

8.8.3.3 loop()

void yaodaq::WebsocketServer::loop ()

Definition at line 107 of file [WebsocketServer.cpp](#).

```
00108 {
00109     listen();
00110     start();
00111     m_Looper.supressInstance();
00112     onRaisingSignal();
00113 }
```

8.8.3.4 setVerbosity()

void yaodaq::WebsocketServer::setVerbosity (
const yaodaq::LoggerHandler::Verbosity & verbosity)

Definition at line 99 of file [WebsocketServer.cpp](#).

```
00099 { m_Logger.setVerbosity( verbosity ); }
```

8.8.3.5 start()

void yaodaq::WebsocketServer::start ()

Definition at line 78 of file [WebsocketServer.cpp](#).

```
00079 {
00080     if( !m_isStarted )
00081     {
00082         m_isStarted = true;
00083         logger()->trace( "Server started" );
00084         ix::WebSocketServer::start();
00085     }
00086 }
```

8.8.3.6 stop()

void yaodaq::WebsocketServer::stop (
bool useless = true)

Definition at line 88 of file [WebsocketServer.cpp](#).

```
00089 {
00090     if( !m_isStopped )
00091     {
00092         m_isStopped = true;
00093         useless      = !useless;
00094         logger()->trace( "Server stopped" );
00095     }
```

```
00095     ix::WebSocketServer::stop();  
00096   }  
00097 }
```

The documentation for this class was generated from the following files:

- yaodaq/[WebSocketServer.hpp](#)
- yaodaq/[WebSocketServer.cpp](#)

Chapter 9

File Documentation

9.1 docs/License.md File Reference

9.2 docs/Third-party licenses.md File Reference

9.3 yaodag/Class.hpp File Reference

```
#include <cstdint>
```

Namespaces

- namespace [yaodag](#)

Enumerations

- enum class [yaodag::Class](#) : std::int_least16_t {
 [yaodag::Unknown](#) = -1 , [yaodag::Module](#) = 0 , [yaodag::Browser](#) = 100 , [yaodag::WebsocketServer](#) = Module
 + 1 ,
 [yaodag::WebsocketClient](#) = Module + 2 }

9.4 Class.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAG_CLASS_HPP
00002 #define YAODAG_CLASS_HPP
00003
00008 #include <cstdint>
00009
00010 namespace yaodag
00011 {
00012
00013 enum class Class : std::int_least16_t
00014 {
00015     Unknown = -1,
00016     Module = 0,
00017     Browser = 100,
00018
00019     WebsocketServer = Module + 1,
00020     WebsocketClient = Module + 2,
00021 };
00022
00023 } // namespace yaodag
00024
00025 #endif // YAODAG_CLASS_HPP
```

9.5 yaodaq/Exception.hpp File Reference

```
#include <cstdint>
#include <exception>
#include <fmt/color.h>
#include <source_location/source_location.hpp>
#include <string>
```

Data Structures

- class [yaodaq::Exception](#)

Namespaces

- namespace [yaodaq](#)

9.6 Exception.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_EXCEPTION
00002 #define YAODAQ_EXCEPTION
00003
00008 #include <cstdint>
00009 #include <exception>
00010 #include <fmt/color.h>
00011 #include <source_location/source_location.hpp>
00012 #include <string>
00013
00014 namespace yaodaq
00015 {
00016
00017     enum class StatusCode : std::int_least32_t;
00018
00019     class Exception : public std::exception, public source_location
00020     {
00021     public:
00022         Exception() = delete;
00023
00024         static void setFormat( const std::string& format ) { m_Format = format; }
00025
00026         static void setStyle( const fmt::text_style& style = {} ) { m_Style = style; }
00027
00028         Exception( const StatusCode& statusCode, const std::string& description, const source_location&
location = source_location::current() );
00029         ~Exception() noexcept override = default;
00030         [[nodiscard]] const char* what() const noexcept final;
00031         [[nodiscard]] const char* description() const noexcept;
00032         [[nodiscard]] std::int_least32_t code() const noexcept;
00033
00034     private:
00035         static fmt::text_style m_Style;
00036         static std::string m_Format;
00037         const std::int_least32_t m_Code{ 0 };
00038         std::string m_Description;
00039         std::string m_Message;
00040         void constructMessage();
00041     };
00042
00043 } // namespace yaodaq
00044
00045 #endif
```

9.7 yaodaq/Identifier.hpp File Reference

```
#include "yaodaq/Class.hpp"
#include <string>
```

Data Structures

- class [yaodaq::Identifier](#)

Namespaces

- namespace [yaodaq](#)

9.8 Identifier.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_IDENTIFIER_HPP
00002 #define YAODAQ_IDENTIFIER_HPP
00003
00008 #include "yaodaq/Class.hpp"
00009
00010 #include <string>
00011
00012 namespace yaodaq
00013 {
00014
00015 class Identifier
00016 {
00017 public:
00018     Identifier() = default;
00019     Identifier( const Class& aClass, const std::string& type, const std::string& name );
00020     [[nodiscard]] std::string getClass() const;
00021     [[nodiscard]] std::string getType() const;
00022     [[nodiscard]] std::string getName() const;
00023     [[nodiscard]] Class getClassId() const;
00024     [[nodiscard]] std::string get() const;
00025
00026 private:
00027     Class m_Class{ Class::Unknown };
00028     std::string m_Type{ "Unknown" };
00029     std::string m_Name{ "Unknown" };
00030 };
00031
00032 } // namespace yaodaq
00033
00034 #endif // YAODAQ_IDENTIFIER_HPP
```

9.9 yaodaq/Interrupt.hpp File Reference

```
#include "yaodaq/Signal.hpp"
#include <atomic>
#include <csignal>
#include <mutex>
```

Data Structures

- class [yaodaq::Interrupt](#)

Namespaces

- namespace [yaodaq](#)

9.10 Interrupt.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_HANDLER_HPP
00002 #define YAODAQ_HANDLER_HPP
00003
00008 #include "yaodaq/Signal.hpp"
00009
00010 #include <atomic>
00011 #include <csignal>
00012 #include <mutex>
```

```

00013
00014 namespace yaodaq
00015 {
00016
00017 enum class Signal;
00018
00019 class Interrupt
00020 {
00021 public:
00022     Interrupt();
00023     void init();
00024     void restore();
00025     Signal getSignal();
00026     ~Interrupt();
00027
00028 private:
00029     volatile static std::atomic<Signal> m_Signal;
00030     void setSignal( const Signal& signal );
00031     std::mutex m_mutex;
00032 };
00033
00034 } // namespace yaodaq
00035
00036 #endif // YAODAQ_HANDLER_HPP

```

9.11 yaodaq/LoggerHandler.hpp File Reference

```

#include <memory>
#include <spdlog/fwd.h>
#include <string>
#include <vector>

```

Data Structures

- class [yaodaq::LoggerHandler](#)

Namespaces

- namespace [spdlog](#)
- namespace [yaodaq](#)

Typedefs

- using [spdlog::sink_ptr](#) = std::shared_ptr< spdlog::sinks::sink >

9.12 LoggerHandler.hpp

[Go to the documentation of this file.](#)

```

00001 #ifndef YAODAQ_LOGGERHANDLER
00002 #define YAODAQ_LOGGERHANDLER
00003
00008 #include <memory>
00009 #include <spdlog/fwd.h>
00010 #include <string>
00011 #include <vector>
00012
00013 namespace spdlog
00014 {
00015     using sink_ptr = std::shared_ptr<spdlog::sinks::sink>;
00016 }
00017
00018 namespace yaodaq
00019 {
00020
00021 class LoggerHandler
00022 {
00023 public:
00024     enum class Verbosity
00025     {
00026         Off,
00027         Trace,

```



```

00028     Debug,
00029     Info,
00030     Warn,
00031     Error,
00032     Critical
00033 };
00034 LoggerHandler();
00035 explicit LoggerHandler( const std::string& );
00036 ~LoggerHandler();
00037 void          setVerbosity( const Verbosity& verbosity );
00038 std::shared_ptr<spdlog::logger> logger();
00039 void          addSink( const spdlog::sink_ptr& );
00040 void          clearSinks();
00041
00042 private:
00043     std::shared_ptr<spdlog::logger> m_Logger{ nullptr };
00044     std::vector<spdlog::sink_ptr> m_Sinks;
00045     std::string m_Name{ "Unknown" };
00046     Verbosity m_Verbosity{ Verbosity::Trace };
00047     void init();
00048 };
00049
00050 } // namespace yaodaq
00051
00052 #endif

```

9.13 yaodaq/Looper.hpp File Reference

```
#include "yaodaq/Interrupt.hpp"
```

Data Structures

- class [yaodaq::Looper](#)

Namespaces

- namespace [yaodaq](#)

9.14 Looper.hpp

[Go to the documentation of this file.](#)

```

00001 #ifndef YAODAQ_LOOPER
00002 #define YAODAQ_LOOPER
00003
00008 #include "yaodaq/Interrupt.hpp"
00009
00010 namespace yaodaq
00011 {
00012
00013     enum class Signal;
00014
00015     class Looper
00016     {
00017     public:
00018         Looper();
00019         Signal loop();
00020         Signal getSignal();
00021         static int getInstance();
00022         void supressInstance();
00023         ~Looper();
00024
00025     private:
00026         static int m_instance;
00027         bool m_hasBeenAdded{ false };
00028         bool m_hasBeenSupressed{ false };
00029         static Interrupt m_interrupt;
00030     };
00031
00032 } // namespace yaodaq
00033
00034 #endif // YAODAQ_LOOPER

```

9.15 yaodaq/Severity.hpp File Reference

```
#include <cstdint>
```

Namespaces

- namespace [yaodaq](#)

Enumerations

- enum class [yaodaq::Severity](#) : std::int_least16_t { [yaodaq::Info](#) = 1 , [yaodaq::Warning](#) = 10 , [yaodaq::Error](#) = 100 , [yaodaq::Critical](#) = 1000 }

9.16 Severity.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_SEVERITY
00002 #define YAODAQ_SEVERITY
00003
00004 #include <cstdint>
00005
00010 namespace yaodaq
00011 {
00012
00013 enum class Severity : std::int_least16_t
00014 {
00015     Info      = 1,
00016     Warning   = 10,
00017     Error     = 100,
00018     Critical  = 1000,
00019 };
00020
00021 } // namespace yaodaq
00022
00023 #endif // YAODAQ_SEVERITY
```

9.17 yaodaq/Signal.hpp File Reference

```
#include "yaodaq/Severity.hpp"
#include <cstdint>
```

Namespaces

- namespace [yaodaq](#)

Enumerations

- enum class [yaodaq::Signal](#) { [yaodaq::NO](#) = 0 , [yaodaq::ABRT](#) = static_cast<int>(Severity::Critical) + 1 , [yaodaq::FPE](#) = static_cast<int>(Severity::Critical) + 2 , [yaodaq::ILL](#) = static_cast<int>(Severity::Critical) + 3 , [yaodaq::SEGV](#) = static_cast<int>(Severity::Critical) + 4 , [yaodaq::INT](#) = static_cast<int>(Severity::Warning) + 1 , [yaodaq::TERM](#) = static_cast<int>(Severity::Warning) + 2 }

9.18 Signal.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_SIGNAL
00002 #define YAODAQ_SIGNAL
00003
00008 #include "yaodaq/Severity.hpp"
00009
00010 #include <cstdint>
00011
```

```

00012 namespace yaodaq
00013 {
00014
00015 enum class Signal
00016 {
00017     NO = 0, // No Signal.
00018     // Critical
00019     ABRT = static_cast<int>( Severity::Critical ) + 1, // (Signal Abort) Abnormal termination, such as
               is initiated by the abort function.
00020     FPE = static_cast<int>( Severity::Critical ) + 2, // (Signal Floating-Point Exception) Erroneous
               arithmetic operation, such as zero divide or an operation resulting in overflow (not necessarily with
               a floating-point operation).
00021     ILL = static_cast<int>( Severity::Critical ) + 3, // (Signal Illegal Instruction) Invalid function
               image, such as an illegal instruction. This is generally due to a corruption in the code or to an
               attempt to execute data.
00022     SEGV = static_cast<int>( Severity::Critical ) + 4, // (Signal Segmentation Violation) Invalid
               access to storage: When a program tries to read or write outside the memory it has allocated.
00023     // Warning
00024     INT = static_cast<int>( Severity::Warning ) + 1, // (Signal Interrupt) Interactive attention
               signal. Generally generated by the application user.
00025     TERM = static_cast<int>( Severity::Warning ) + 2, // (Signal Terminate) Termination request sent to
               program.
00026 };
00027
00028 } // namespace yaodaq
00029
00030 #endif // YAODAQ_CLASS_HPP

```

9.19 yaodaq/StatusCode.hpp File Reference

```
#include <cstdint>
```

Namespaces

- namespace `yaodaq`

Enumerations

- enum class `yaodaq::StatusCode` : `std::int_least32_t` { `yaodaq::SUCCESS` = 0 , `yaodaq::LISTEN_ERROR` }

9.20 StatusCode.hpp

[Go to the documentation of this file.](#)

```

00001 #ifndef YAODAQ_STATUSCODE
00002 #define YAODAQ_STATUSCODE
00003
00008 #include <cstdint>
00009
00010 namespace yaodaq
00011 {
00012
00013 enum class StatusCode : std::int_least32_t
00014 {
00015     SUCCESS = 0,
00016     LISTEN_ERROR,
00017 };
00018
00019 }
00020
00021 #endif

```

9.21 yaodaq/Version.hpp File Reference

```

#include <cstdint>
#include <semver.hpp>
#include <string>

```

Data Structures

- class [yaodaq::Version](#)

Namespaces

- namespace [yaodaq](#)

9.22 Version.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_VERSION_HPP
00002 #define YAODAQ_VERSION_HPP
00003
00008 #include <stdint>
00009 #include <semver.hpp>
00010 #include <string>
00011
00012 namespace yaodaq
00013 {
00014
00015 class Version : public semver::version
00016 {
00017 public:
00018     constexpr Version( const std::uint8_t& mj, const std::uint8_t& mn, const std::uint8_t& pt, const
semver::prerelease& prt = semver::prerelease::none, const std::uint8_t& prn = 0 ) noexcept :
semver::version( mj, mn, pt, prt, prn ) {}
00019     explicit constexpr Version( const std::string_view& str ) : semver::version( str ) {}
00020     constexpr Version() = default;
00021     std::uint8_t getMajor();
00022     std::uint8_t getMinor();
00023     std::uint8_t getPatch();
00024     std::string getPreRelease();
00025     std::uint8_t getPreReleaseNumber();
00026 };
00027
00028 } // namespace yaodaq
00029
00030 #endif // YAODAQ_VERSION_HPP
```

9.23 yaodaq/WebsocketClient.hpp File Reference

```
#include "yaodaq/Identifier.hpp"
#include "yaodaq/Interrupt.hpp"
#include "yaodaq/LoggerHandler.hpp"
#include "yaodaq/Looper.hpp"
#include "yaodaq/YaodaqVersion.hpp"
#include <ixwebsocket/IXWebSocket.h>
#include <memory>
#include <spdlog/spdlog.h>
#include <string>
```

Data Structures

- class [yaodaq::WebsocketClient](#)

Namespaces

- namespace [yaodaq](#)

9.24 WebsocketClient.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef YAODAQ_WEBSOCKETCLIENT
00002 #define YAODAQ_WEBSOCKETCLIENT
00003
```

```

00008 #include "yaodaq/Identifier.hpp"
00009 #include "yaodaq/Interrupt.hpp"
00010 #include "yaodaq/LoggerHandler.hpp"
00011 #include "yaodaq/Looper.hpp"
00012 #include "yaodaq/YaodaqVersion.hpp"
00013
00014 #include <ixwebsocket/IXWebSocket.h>
00015 #include <memory>
00016 #include <spdlog/spdlog.h>
00017 #include <string>
00018
00019 namespace yaodaq
00020 {
00021
00022 class WebsocketClient : public ix::WebSocket
00023 {
00024 public:
00025     explicit WebsocketClient( const std::string& name, const std::string& type =
        "DefaultWebsocketClient" );
00026     virtual ~WebsocketClient();
00027     void start();
00028     void stop();
00029     void loop();
00030     std::shared_ptr<spdlog::logger> logger() { return m_Logger.logger(); }
00031
00032 private:
00033     void onRaisingSignal();
00034     Identifier m_Identifier;
00035     LoggerHandler m_Logger;
00036     Looper m_Looper;
00037 };
00038
00039 } // namespace yaodaq
00040
00041 #endif

```

9.25 yaodaq/WebsocketServer.hpp File Reference

```

#include "yaodaq/Identifier.hpp"
#include "yaodaq/Interrupt.hpp"
#include "yaodaq/LoggerHandler.hpp"
#include "yaodaq/Looper.hpp"
#include "yaodaq/YaodaqVersion.hpp"
#include <ixwebsocket/IXWebSocketServer.h>
#include <memory>
#include <spdlog/spdlog.h>
#include <string>

```

Data Structures

- class [yaodaq::WebsocketServer](#)

Namespaces

- namespace [yaodaq](#)

9.26 WebsocketServer.hpp

[Go to the documentation of this file.](#)

```

00001 #ifndef YAODAQ_WEBSOCKETSERVER
00002 #define YAODAQ_WEBSOCKETSERVER
00003
00008 #include "yaodaq/Identifier.hpp"
00009 #include "yaodaq/Interrupt.hpp"
00010 #include "yaodaq/LoggerHandler.hpp"
00011 #include "yaodaq/Looper.hpp"
00012 #include "yaodaq/YaodaqVersion.hpp"
00013
00014 #include <ixwebsocket/IXWebSocketServer.h>
00015 #include <memory>
00016 #include <spdlog/spdlog.h>

```

```

00017 #include <string>
00018
00019 namespace yaodag
00020 {
00021
00022 class WebsocketServer : public ix::WebsocketServer
00023 {
00024 public:
00025     explicit WebsocketServer( const std::string& name, const int& port = ix::SocketServer::kDefaultPort,
00026                             const std::string& host = ix::SocketServer::kDefaultHost, const int& backlog =
00027                             ix::SocketServer::kDefaultTcpBacklog,
00028                             const std::size_t& maxConnections =
00029                             ix::SocketServer::kDefaultMaxConnections, const int& handshakeTimeoutSecs =
00030                             ix::WebsocketServer::kDefaultHandShakeTimeoutSecs, const int& addressFamily =
00031                             ix::SocketServer::kDefaultAddressFamily,
00032                             const std::string& type = "DefaultWebsocketServer" );
00033     virtual ~WebsocketServer();
00034     void loop();
00035     void start();
00036     void stop( bool useless = true );
00037     void listen();
00038
00039     void setVerbosity( const yaodag::LoggerHandler::Verbosity& verbosity );
00040
00041     std::shared_ptr<spdlog::logger> logger() { return m_Logger.logger(); }
00042 private:
00043     void onRaisingSignal();
00044     bool m_isListening{ false };
00045     Identifier m_Identifier;
00046     LoggerHandler m_Logger;
00047     Interrupt m_Interrupt;
00048     Looper m_Looper;
00049     bool m_isStopped{ false };
00050     bool m_isStarted{ false };
00051 };
00052 } // namespace yaodag
00053
00054 #ifdef YAODAG_WEBSOCKETSERVER

```

9.27 yaodag/Exception.cpp File Reference

```
#include "yaodag/Exception.hpp"
```

Namespaces

- namespace `yaodag`

9.28 Exception.cpp

[Go to the documentation of this file.](#)

```

00001
00002 #include "yaodag/Exception.hpp"
00003
00004 namespace yaodag
00005 {
00006
00007 std::string Exception::m_Format{ "\n\t[Code] : {Code}\n\t[Description] : {Description}\n\t[File] :
00008 {File}\n\t[Function] : {Function}\n\t[Line] : {Line}\n\t[Column] : {Column}\n" };
00009
00010 fmt::text_style Exception::m_Style = { fg( fmt::color::crimson ) | fmt::emphasis::bold };
00011
00012 Exception::Exception( const StatusCode& statusCode, const std::string& description, const
00013 source_location& location ) : source_location( location ), m_Code( static_cast<std::int_least32_t>(
00014 statusCode ) ), m_Description( description ) { constructMessage(); }
00015
00016 const char* Exception::what() const noexcept { return m_Message.c_str(); }
00017
00018 const char* Exception::description() const noexcept { return m_Description.c_str(); }
00019
00020 int_least32_t Exception::code() const noexcept { return m_Code; }
00021
00022 void Exception::constructMessage()
00023 {
00024     m_Message = fmt::format( m_Style, m_Format, fmt::arg( "Code", m_Code ), fmt::arg( "Description",
00025 m_Description ), fmt::arg( "File", file_name() ), fmt::arg( "Function", function_name() ), fmt::arg(
00026 "Column", column() ), fmt::arg( "Line", line() ) );
00027 }
00028

```

```

00025 }
00026
00027 } // namespace yaodaq

```

9.29 yaodaq/Identifier.cpp File Reference

```

#include "yaodaq/Identifier.hpp"
#include "yaodaq/Class.hpp"
#include <fmt/color.h>
#include <magic_enum.hpp>
#include <string>

```

Namespaces

- namespace [yaodaq](#)

9.30 Identifier.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/Identifier.hpp"
00006
00007 #include "yaodaq/Class.hpp"
00008
00009 #include <fmt/color.h>
00010 #include <magic_enum.hpp>
00011 #include <string>
00012
00013 namespace yaodaq
00014 {
00015
00016 Identifier::Identifier( const Class& aClass, const std::string& type, const std::string& name ) :
    m_Class( aClass ), m_Type( type ), m_Name( name ) {}
00017
00018 std::string Identifier::getClass() const { return std::string( magic_enum::enum_name( m_Class ) ); }
00019
00020 std::string Identifier::getType() const { return m_Type; }
00021
00022 std::string Identifier::getName() const { return m_Name; }
00023
00024 Class Identifier::getClassId() const { return m_Class; }
00025
00026 std::string Identifier::get() const { return fmt::format( "{0}/{1}/{2}", getClass(), getType(),
    getName() ); }
00027
00028 } // namespace yaodaq

```

9.31 yaodaq/Interrupt.cpp File Reference

```

#include "yaodaq/Interrupt.hpp"
#include "yaodaq/Signal.hpp"
#include <atomic>
#include <csignal>
#include <mutex>
#include <thread>

```

Namespaces

- namespace [yaodaq](#)

9.32 Interrupt.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/Interrupt.hpp"
00006
00007 #include "yaodaq/Signal.hpp"
00008
00009 #include <atomic>
00010 #include <csignal>
00011 #include <mutex>
00012 #include <thread>
00013
00014 namespace yaodaq
00015 {
00016
00017 volatile std::atomic<Signal> Interrupt::m_Signal = Signal::NO;
00018
00019 Interrupt::Interrupt() { init(); }
00020
00021 void Interrupt::restore()
00022 {
00023     std::signal( SIGTERM, SIG_DFL );
00024     std::signal( SIGSEGV, SIG_DFL );
00025     std::signal( SIGINT, SIG_DFL );
00026     std::signal( SIGILL, SIG_DFL );
00027     std::signal( SIGABRT, SIG_DFL );
00028     std::signal( SIGFPE, SIG_DFL );
00029 }
00030
00031 void Interrupt::init()
00032 {
00033     setSignal( Signal::TERM );
00034     setSignal( Signal::TERM );
00035     setSignal( Signal::SEGV );
00036     setSignal( Signal::INT );
00037     setSignal( Signal::ILL );
00038     setSignal( Signal::ABRT );
00039     setSignal( Signal::FPE );
00040 }
00041
00042 Interrupt::~Interrupt() { restore(); }
00043
00044 Signal Interrupt::getSignal()
00045 {
00046     if( m_Signal.load() != Signal::NO )
00047     {
00048         std::lock_guard<std::mutex> guard( m_mutex );
00049         init();
00050     }
00051     return m_Signal.load();
00052 }
00053
00054 void Interrupt::setSignal( const Signal& signal )
00055 {
00056     switch( signal )
00057     {
00058         case Signal::ABRT: std::signal( SIGABRT, []( int ) -> void { m_Signal.store( Signal::ABRT ); } );
00059         break;
00060         case Signal::FPE: std::signal( SIGFPE, []( int ) -> void { m_Signal.store( Signal::FPE ); } );
00061         break;
00062         case Signal::ILL: std::signal( SIGILL, []( int ) -> void { m_Signal.store( Signal::ILL ); } );
00063         break;
00064         case Signal::SEGV: std::signal( SIGSEGV, []( int ) -> void { m_Signal.store( Signal::SEGV ); } );
00065         break;
00066         case Signal::INT: std::signal( SIGINT, []( int ) -> void { m_Signal.store( Signal::INT ); } );
00067         break;
00068         case Signal::TERM: std::signal( SIGTERM, []( int ) -> void { m_Signal.store( Signal::TERM ); } );
00069         break;
00070         default: break;
00071     }
00072 }
00073 } // namespace yaodaq

```

9.33 yaodaq/LoggerHandler.cpp File Reference

```

#include "yaodaq/LoggerHandler.hpp"
#include "spdlog/spdlog.h"

```

Namespaces

- namespace [yaodaq](#)

9.34 LoggerHandler.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/LoggerHandler.hpp"
00006
00007 #include "spdlog/spdlog.h"
00008
00009 namespace yaodaq
00010 {
00011
00012 LoggerHandler::LoggerHandler() { init(); }
00013
00014 LoggerHandler::LoggerHandler( const std::string& name ) : m_Name( name ) { init(); }
00015
00016 LoggerHandler::~LoggerHandler() {}
00017
00018 void LoggerHandler::setVerbosity( const Verbosity& verbosity )
00019 {
00020     m_Verbosity = verbosity;
00021     init();
00022 }
00023
00024 void LoggerHandler::init()
00025 {
00026     m_Logger = std::make_shared<spdlog::logger>( m_Name, std::begin( m_Sinks ), std::end( m_Sinks ) );
00027     switch( m_Verbosity )
00028     {
00029         case Verbosity::Off: m_Logger->set_level( spdlog::level::off ); break;
00030         case Verbosity::Trace: m_Logger->set_level( spdlog::level::trace ); break;
00031         case Verbosity::Debug: m_Logger->set_level( spdlog::level::debug ); break;
00032         case Verbosity::Info: m_Logger->set_level( spdlog::level::info ); break;
00033         case Verbosity::Warn: m_Logger->set_level( spdlog::level::warn ); break;
00034         case Verbosity::Error: m_Logger->set_level( spdlog::level::err ); break;
00035         case Verbosity::Critical: m_Logger->set_level( spdlog::level::critical ); break;
00036     }
00037 }
00038
00039 std::shared_ptr<spdlog::logger> LoggerHandler::logger() { return std::shared_ptr<spdlog::logger>(
    m_Logger ); }
00040
00041 void LoggerHandler::addSink( const spdlog::sink_ptr& sink )
00042 {
00043     m_Sinks.push_back( sink );
00044     init();
00045 }
00046
00047 void LoggerHandler::clearSinks()
00048 {
00049     m_Sinks.clear();
00050     init();
00051 }
00052
00053 } // namespace yaodaq

```

9.35 yaodaq/Looper.cpp File Reference

```

#include "yaodaq/Looper.hpp"
#include <chrono>
#include <thread>

```

Namespaces

- namespace [yaodaq](#)

9.36 Looper.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/Looper.hpp"
00006
00007 #include <chrono>
00008 #include <thread>
00009
00010 namespace yaodaq
00011 {

```

```

00012
00013 int Looper::m_instance{ 0 };
00014
00015 Interrupt Looper::m_Interrupt{ Interrupt{} };
00016
00017 int Looper::getInstance() { return m_instance; }
00018
00019 void Looper::supressInstance()
00020 {
00021     if( m_hasBeenSupressed == false )
00022     {
00023         m_hasBeenSupressed = true;
00024         m_instance--;
00025     }
00026 }
00027
00028 Looper::Looper()
00029 {
00030     if( m_hasBeenAdded == false )
00031     {
00032         m_hasBeenAdded = true;
00033         ++m_instance;
00034     }
00035 }
00036
00037 Signal Looper::loop()
00038 {
00039     static Signal signal{ yaodaq::Signal::NO };
00040     if( m_instance == 0 )
00041     {
00042         do {
00043             signal = m_Interrupt.getSignal();
00044             std::this_thread::sleep_for( std::chrono::microseconds( 1 ) );
00045         } while( signal == yaodaq::Signal::NO );
00046     }
00047     return signal;
00048 }
00049
00050 Signal Looper::getSignal() { return m_Interrupt.getSignal(); }
00051
00052 Looper::~~Looper()
00053 {
00054     if( m_hasBeenAdded == true && m_hasBeenSupressed == false )
00055     {
00056         m_hasBeenSupressed = true;
00057         --m_instance;
00058     }
00059 }
00060
00061 } // namespace yaodaq

```

9.37 yaodaq/Version.cpp File Reference

```

#include "yaodaq/Version.hpp"
#include <magic_enum.hpp>

```

Namespaces

- namespace [yaodaq](#)

9.38 Version.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/Version.hpp"
00006
00007 #include <magic_enum.hpp>
00008
00009 namespace yaodaq
00010 {
00011
00012 std::uint8_t Version::getMajor() { return major; }
00013
00014 std::uint8_t Version::getMinor() { return minor; }
00015
00016 std::uint8_t Version::getPatch() { return patch; }
00017

```

```

00018 std::string Version::getPreRelease() { return std::string( magic_enum::enum_name( prerelease_type ) );
    }
00019
00020 std::uint8_t Version::getPreReleaseNumber() { return prerelease_number; }
00021
00022 const static Version yaodaq_version;
00023
00024 } // namespace yaodaq

```

9.39 yaodaq/WebsocketClient.cpp File Reference

```

#include "yaodaq/WebsocketClient.hpp"
#include <chrono>
#include <ixwebsocket/IXNetSystem.h>
#include <magic_enum.hpp>
#include <spdlog/sinks/stdout_color_sinks.h>
#include <thread>

```

Namespaces

- namespace [yaodaq](#)

9.40 WebsocketClient.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/WebsocketClient.hpp"
00006
00007 #include <chrono>
00008 #include <ixwebsocket/IXNetSystem.h>
00009 #include <magic_enum.hpp>
00010 #include <spdlog/sinks/stdout_color_sinks.h>
00011 #include <thread>
00012
00013 namespace yaodaq
00014 {
00015
00016 WebsocketClient::WebsocketClient( const std::string& name, const std::string& type ) : m_Identifier(
    Class::WebsocketClient, type, name ), m_Logger( m_Identifier.get() )
00017 {
00018     ix::initNetSystem();
00019     ix::WebSocketHttpHeaders header{ { "Id", m_Identifier.get() } };
00020     setExtraHeaders( header );
00021     m_Logger.addSink( std::make_shared<spdlog::sinks::stdout_color_sink_mt>() );
00022     setOnMessageCallback(
00023         [this]( const ix::WebSocketMessagePtr& msg )
00024         {
00025             if( msg->type == ix::WebSocketMessageType::Message ) { logger()->error( "{}", msg->str ); }
00026         } );
00027 }
00028
00029 WebsocketClient::~WebsocketClient()
00030 {
00031     stop();
00032     ix::uninitNetSystem();
00033 }
00034
00035 void WebsocketClient::start()
00036 {
00037     if( getReadyState() == ix::ReadyState::Closed || getReadyState() == ix::ReadyState::Closing )
00038     {
00039         logger()->trace( "Client started. Connected to {}", getUrl() );
00040         ix::WebSocket::start();
00041     }
00042 }
00043
00044 void WebsocketClient::stop()
00045 {
00046     if( getReadyState() == ix::ReadyState::Open || getReadyState() == ix::ReadyState::Connecting )
00047     {
00048         logger()->trace( "Client stopped" );
00049         ix::WebSocket::stop();
00050         while( getReadyState() != ix::ReadyState::Closed ) { std::this_thread::sleep_for(
            std::chrono::microseconds( 1 ) ); }
00051     }

```

```

00052 }
00053
00054 void WebSocketClient::loop()
00055 {
00056     WebSocketClient::start();
00057     m_Looper.supressInstance();
00058     onRaisingSignal();
00059 }
00060
00061 void WebSocketClient::onRaisingSignal()
00062 {
00063     Signal signal = m_Looper.loop();
00064     if( m_Looper.getInstance() == 0 )
00065     {
00066         int value = magic_enum::enum_integer( signal );
00067         if( value >= magic_enum::enum_integer( yaodaq::Severity::Critical ) ) { logger()->critical(
00068             "Signal SIG{} raised !", magic_enum::enum_name( signal ) ); }
00069         else if( value >= magic_enum::enum_integer( yaodaq::Severity::Error ) )
00070         {
00071             logger()->error( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00072         }
00073         else if( value >= magic_enum::enum_integer( yaodaq::Severity::Warning ) )
00074         {
00075             fmt::print( "\n" );
00076             logger()->warn( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00077         }
00078         else if( value >= magic_enum::enum_integer( yaodaq::Severity::Info ) )
00079         {
00080             fmt::print( "\n" );
00081             logger()->info( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00082         }
00083         else
00084         {
00085             fmt::print( "\n" );
00086             logger()->trace( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00087         }
00088         if( magic_enum::enum_integer( signal ) >= magic_enum::enum_integer( Severity::Critical ) )
00089             std::exit( magic_enum::enum_integer( signal ) );
00090     }
00091 } // namespace yaodaq

```

9.41 yaodaq/WebsocketServer.cpp File Reference

```

#include "yaodaq/WebsocketServer.hpp"
#include "yaodaq/Exception.hpp"
#include "yaodaq/StatusCode.hpp"
#include <iostream>
#include <ixwebsocket/IXNetSystem.h>
#include <magic_enum.hpp>
#include <spdlog/sinks/stdout_color_sinks.h>
#include <spdlog/spdlog.h>
#include <string>
#include <thread>
#include <utility>

```

Namespaces

- namespace [yaodaq](#)

9.42 WebsocketServer.cpp

[Go to the documentation of this file.](#)

```

00001
00005 #include "yaodaq/WebsocketServer.hpp"
00006
00007 #include "yaodaq/Exception.hpp"
00008 #include "yaodaq/StatusCode.hpp"
00009
00010 #include <iostream>
00011 #include <ixwebsocket/IXNetSystem.h>

```

```

00012 #include <magic_enum.hpp>
00013 #include <spdlog/sinks/stdout_color_sinks.h>
00014 #include <spdlog/spdlog.h>
00015 #include <string>
00016 #include <thread>
00017 #include <utility>
00018
00019 namespace yaodan
00020 {
00021
00022 WebSocketServer::WebSocketServer( const std::string& name, const int& port, const std::string& host,
    const int& backlog, const std::size_t& maxConnections, const int& handshakeTimeoutSecs, const int&
    addressFamily, const std::string& type ) :
00023     ix::WebSocketServer( port, host, backlog, maxConnections, handshakeTimeoutSecs, addressFamily ),
    m_Identifier( Class::WebSocketServer, type, name ), m_Logger( m_Identifier.get() )
00024 {
00025     ix::initNetSystem();
00026     m_Logger.addSink( std::make_shared<spdlog::sinks::stdout_color_sink_mt>() );
00027     setOnClientMessageCallback(
00028         []( std::shared_ptr<ix::ConnectionState> connectionState, ix::WebSocket& websocket, const
    ix::WebSocketMessagePtr& msg )
00029         {
00030             // The ConnectionState object contains information about the connection,
00031             // at this point only the client ip address and the port.
00032             std::cout << "Remote ip: " << connectionState->getRemoteIp() << std::endl;
00033
00034             if( msg->type == ix::WebSocketMessageType::Open )
00035             {
00036                 std::cout << "New connection" << std::endl;
00037
00038                 // A connection state object is available, and has a default id
00039                 // You can subclass ConnectionState and pass an alternate factory
00040                 // to override it. It is useful if you want to store custom
00041                 // attributes per connection (authenticated bool flag, attributes, etc...)
00042                 std::cout << "id: " << connectionState->getId() << std::endl;
00043
00044                 // The uri the client did connect to.
00045                 std::cout << "Uri: " << msg->openInfo.uri << std::endl;
00046
00047                 std::cout << "Headers:" << std::endl;
00048                 for( auto it: msg->openInfo.headers ) { std::cout << "\t" << it.first << ": " << it.second <<
    std::endl; }
00049             }
00050             else if( msg->type == ix::WebSocketMessageType::Message )
00051             {
00052                 // For an echo server, we just send back to the client whatever was received by the server
00053                 // All connected clients are available in an std::set. See the broadcast cpp example.
00054                 // Second parameter tells whether we are sending the message in binary or text mode.
00055                 // Here we send it in the same mode as it was received.
00056                 std::cout << "Received: " << msg->str << std::endl;
00057
00058                 websocket.send( msg->str, msg->binary );
00059             }
00060         } );
00061 }
00062
00063 void WebSocketServer::listen()
00064 {
00065     if( !m_isListening )
00066     {
00067         std::pair<bool, std::string> ret = ix::WebSocketServer::listen();
00068         if( ret.first )
00069         {
00070             m_isListening = ret.first;
00071             logger()->info( "Server listening on host {0} port {1}", getHost(), getPort() );
00072         }
00073         else
00074             throw Exception( StatusCode::LISTEN_ERROR, ret.second );
00075     }
00076 }
00077
00078 void WebSocketServer::start()
00079 {
00080     if( !m_isStarted )
00081     {
00082         m_isStarted = true;
00083         logger()->trace( "Server started" );
00084         ix::WebSocketServer::start();
00085     }
00086 }
00087
00088 void WebSocketServer::stop( bool useless )
00089 {
00090     if( !m_isStopped )
00091     {
00092         m_isStopped = true;
00093         useless = !useless;

```

```

00094     logger()->trace( "Server stopped" );
00095     ix::WebSocketServer::stop();
00096 }
00097 }
00098
00099 void WebSocketServer::setVerbosity( const yaodag::LoggerHandler::Verbosity& verbosity ) {
    m_Logger.setVerbosity( verbosity ); }
00100
00101 WebSocketServer::~WebSocketServer()
00102 {
00103     stop();
00104     ix::uninitNetSystem();
00105 }
00106
00107 void WebSocketServer::loop()
00108 {
00109     listen();
00110     start();
00111     m_Looper.supressInstance();
00112     onRaisingSignal();
00113 }
00114
00115 void WebSocketServer::onRaisingSignal()
00116 {
00117     Signal signal = m_Looper.loop();
00118     if( m_Looper.getInstance() == 0 )
00119     {
00120         int value = magic_enum::enum_integer( signal );
00121         if( value >= magic_enum::enum_integer( yaodag::Severity::Critical ) ) { logger()->critical(
"Signal SIG{} raised !", magic_enum::enum_name( signal ) ); }
00122         else if( value >= magic_enum::enum_integer( yaodag::Severity::Error ) )
00123         {
00124             logger()->error( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00125         }
00126         else if( value >= magic_enum::enum_integer( yaodag::Severity::Warning ) )
00127         {
00128             fmt::print( "\n" );
00129             logger()->warn( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00130         }
00131         else if( value >= magic_enum::enum_integer( yaodag::Severity::Info ) )
00132         {
00133             fmt::print( "\n" );
00134             logger()->info( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00135         }
00136         else
00137         {
00138             fmt::print( "\n" );
00139             logger()->trace( "Signal SIG{} raised !", magic_enum::enum_name( signal ) );
00140         }
00141         if( magic_enum::enum_integer( signal ) >= magic_enum::enum_integer( Severity::Critical ) )
std::exit( magic_enum::enum_integer( signal ) );
00142     }
00143 }
00144
00145 } // namespace yaodag

```