Environment Toolkit for TestStand

Antidoc v3.1.0, Antidoc CLI, Juan Jose Hasbun Wood <juan@hasbun-wood-engineering.de>

Table of Contents

Chapter 1. Project description

Environment Toolkit for TestStand is an open-source utility that simplifies working with multiple NI TestStand versions and environments on a single system.

It allows users to discover installed TestStand versions, define reusable work profiles (i.e. version with associated environment), and launch profiles easily via a CLI or GUI.

The toolkit is designed for engineers who switch frequently between projects, customers, or TestStand releases and want an efficient way to manage their setups.

Further information can be found in the projects **repository**:

https://github.com/HasbunWoodEngineering/Environment-Toolkit-for-TestStand

This project is licensed under the MIT License. See the **LICENSE** file for details:

https://github.com/HasbunWoodEngineering/Environment-Toolkit-for-TestStand/blob/main/LICENSE

The licenses of all dependencies (runtime and development) are documented in **THIRD_PARTY_LICENSES** and include the full license texts:

https://github.com/HasbunWoodEngineering/Environment-Toolkit-for-TestStand/blob/main/THIRD_PARTY_LICENSES

Trademark Notice

TestStand™ and LabVIEW™ are registered trademarks of National Instruments Corporation.

Environment Toolkit for TestStand is an independent open-source project. It is not affiliated with, endorsed by, or sponsored by National Instruments Corporation.

Any use of the TestStand or LabVIEW names is for descriptive purposes only. All other trademarks are the property of their respective owners.

Chapter 2. DQMH® modules

This section describes DQMH® module responsibilities and relationships.

2.1. Preamble

A DQMH module is the main component of an architecture based on DQMH® framework. A DQMH module is used to implement a section of the application that has one responsibility.

DQMH® framework defines two different type of DQMH module.

Singleton:

A Singleton DQMH module can have only one instance running at any given time.

Cloneable:

A Cloneable DQMH module can have one or multiple instances running in parallel.

DQMH® framework defines two different ways to carry data throughout the application and with both other DQMH modules and non-DQMH based code.

Request events:

A request is a code that fires an event requesting the DQMH module to do something. Multiple locations in the code can send events to the DQMH module.

Request events are many-to-one.

Requests are usually named using imperative tense.

Broadcast events:

A broadcast is a code that fires an event broadcasting that the DQMH module did something. Multiple Event Structures can register to handle the Broadcast Events.

Broadcast Events are one-to-many.

Broadcasts are usually named using past tense or passive voice.

Default events:

By default all the DQMH ® Modules are shipped with some default events (Requests and Broadcasts). Here is the list of this events:

- Start Module
- Stop Module
- Show Panel
- Hide Panel
- Show Diagram
- Status Updated
- Error Reported
- Module Did Stop
- Update Module Execution Status
- Module Did Init
- Get Module Execution Status

If you want to learn more about default DQMH ® Events see the documentaiton.

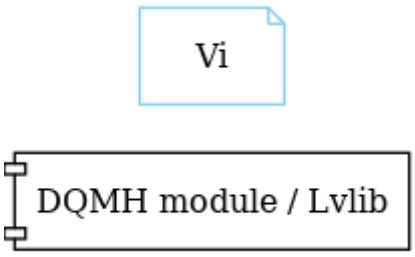
NOTE	Refer to the DQMH® framework official documentation to find more details on how the framework works

The following section gives you details on the project architecture relying on this framework. It gives you an overview of the modules' interaction and detailed information on each module.

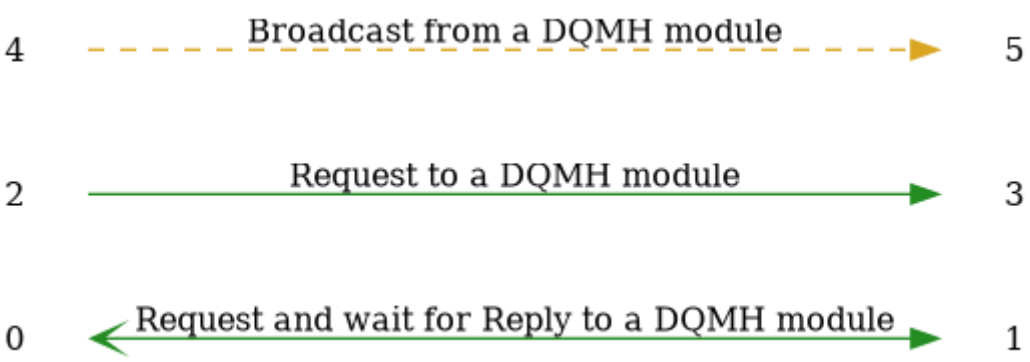
NOTE	If your module has a helper loop, it will be listed along with the events it processes. Helper loops can be categorized as either DQMH or Custom. To ensure your helper loop adheres to DQMH guidelines see the documentaiton.

Graphs used in this section have the following legend:

Components:

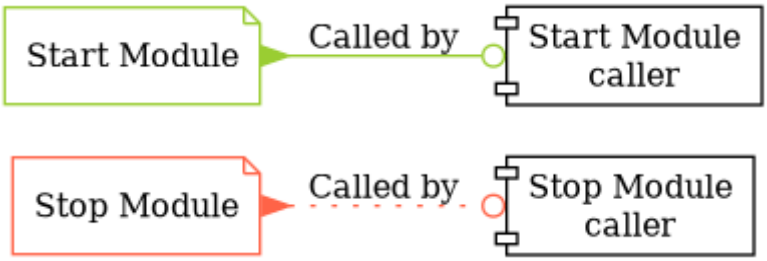


Events:



NOTE	One arrow can represent one or more events between two components
NOTE	Request and Request and wait for Reply are represented by only one arrow. If there is no Request and wait for Reply, Request representation is used. Otherwise Request and wait for Reply is used

Start and Stop module callers:



2.2. Modules overview

This project contains 1 singleton module and 0 cloneable module.

Table 1. Modules list

Singleton	Cloneable
TS Environment Manager.lvlib	

This graph represents the links between all DQMH modules.

2.3. TS Environment Manager.lvlib

Type: Singleton

Responsibility: TS Environment Manager is a DQMH module that centralizes the management and

execution of TestStand profiles.

The module provides functionality to create and manage environment profiles (this refers to a specific TestStand version associated with an environment), activate software versions and launch the Sequence Editor in a controlled and deterministic manner.

2.3.1. Module relationship

Table 2. Requests callers

Request Name	Callers
Activate Version	env-toolkit-ui.vi env-toolkit.vi
Add Profile	env-toolkit-ui.vi env-toolkit.vi
Delete Profile	env-toolkit-ui.vi env-toolkit.vi
Get Installations	env-toolkit-ui.vi env-toolkit.vi
Get Module Execution Status	
Get Profile(s)	env-toolkit-ui.vi env-toolkit.vi
Get Profiles File	TS Environment Manager.lvlib:Main.vi
Hide Panel	
Launch Sequence Editor	env-toolkit-ui.vi env-toolkit.vi
Set Profiles File	env-toolkit-ui.vi env-toolkit.vi
Show Diagram	
Show Panel	

Table 3. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	env-toolkit-ui.vi
Launched SeqEdit	env-toolkit-ui.vi

Broadcast Name	Listeners
Module Did Init	
Module Did Stop	
Status Updated	
Update Module Execution Status	
Version Activated	env-toolkit-ui.vi

Table 4. Used requests

Module	Requests
TS Environment Manager.lvlib	Get Profiles File.vi

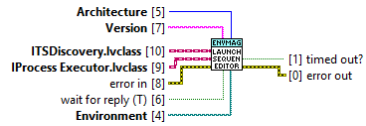
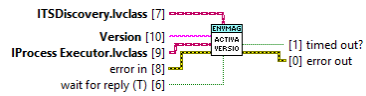
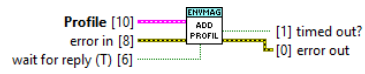
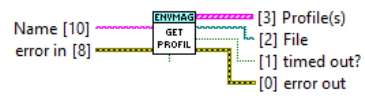
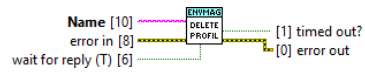
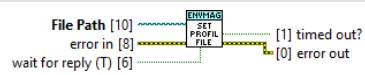
Table 5. Registered broadcast

Module	Broadcasts
—	—

2.3.2. Event list

NOTE The default events are not listed.

Table 6. Events

Name	Type	Connector pane	Description	S.	R.	I.
Get Installations			Retrieves all TS Installations in the system and returns them as an array of TSInstall elements.			
Launch Sequence Editor			Launches the Sequence Editor for the specified TS version optionally with the specified environment.			
Activate Version			Activates the specified TS version.			
Add Profile			Adds a profile to the active file.			
Get Profile(s)			Retrieves the current profile file followed by the specified profile parameters, or all profiles if none is specified.			
Delete Profile			Deletes the specified profile from the current file.			
Set Profiles File			This request sets the specified profiles file in the cache INI file.			

Name	Type	Connector pane	Description	S.	R.	I.
Version Activated		Reply Payload [10] error in [8] [0] error out	This broadcast is sent after a version was successfully activated.			
Launched SeqEdit		Reply Payload [10] error in [8] [0] error out	This broadcast is sent after the sequence editor is launched.			
Get Profiles File		error in [8] [0] error out	Reads the configured profile file location from the cache INI file: *EnvironmentToolkit.ProfilesFile*. If the entry is not found, it is created with the default value *C:\Users\<current-user>\Environment Toolkit Profiles.ini*.			

Type: → Request | → Request and Wait for Reply | → Broadcast

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

2.3.3. Module Start/Stop calls

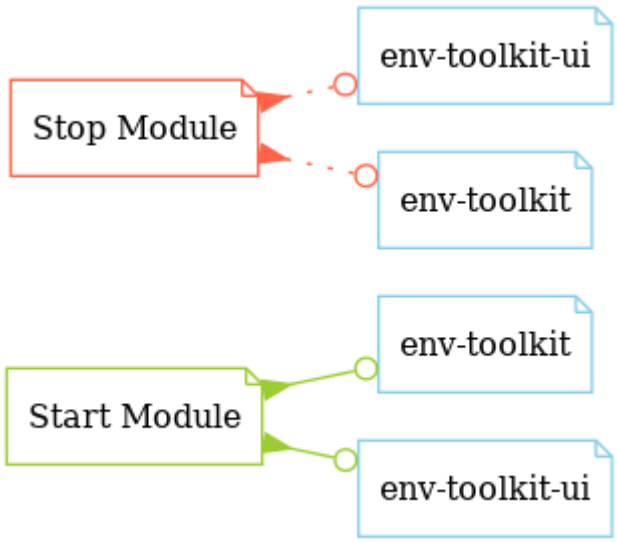


Table 7. Start and Stop module callers

Function	Callers
Start Module	env-toolkit.vi env-toolkit-ui.vi
Stop Module	env-toolkit.vi env-toolkit-ui.vi

2.3.4. Module Helper Loops

NOTE	No Helper Loops Found

2.3.5. Module custom errors

TIP	Custom errors are added to the module via vi named `*--error.vi`.

Module TS Environment Manager.lvlib use the following custom errors:

Table 8. Custom errors

Name	Code	Description
Module Not Running	0	Error information not found in the code
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.
Module Not Synced	403683	%s Module was unable to synchronize events.
Request and Wait for Reply Timeout	403686	

2.3.6. Module Constant VIs

Table 9. Constant VIs Found

VI Name	Data Type	Value
Module Name—constant.vi	String	TS Environment Manager
Module Timeout—constant.vi	I32	5000

Chapter 3. Classes

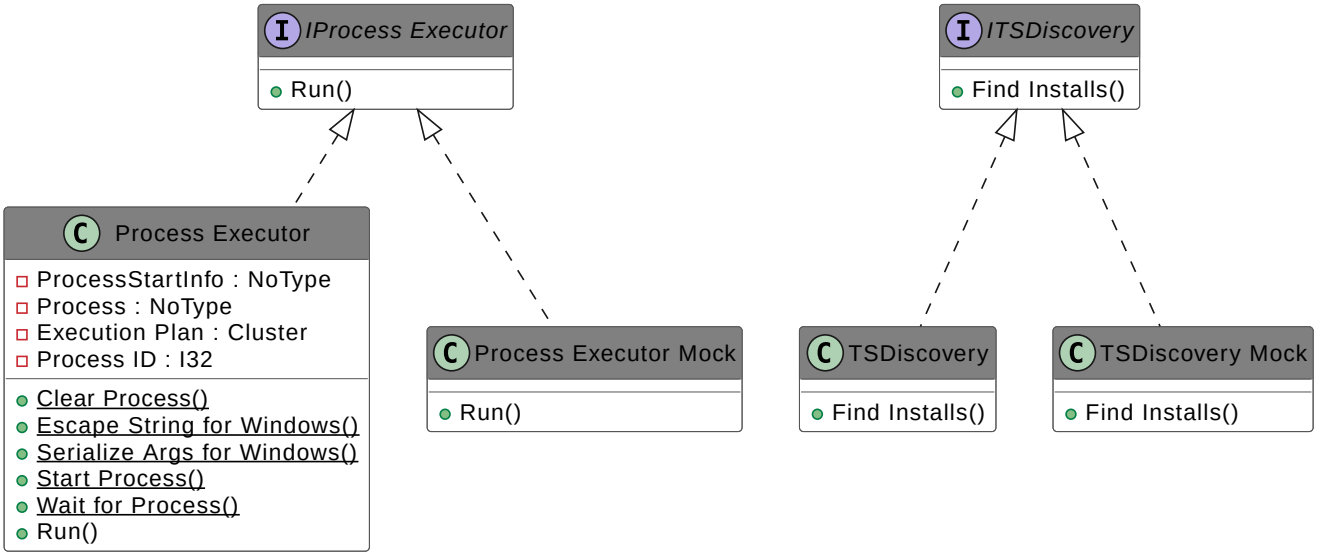
This section describes the classes contained in the project.

3.1. Classes overview

This project contains 4 classes and 2 interfaces.

Table 10. Classes list

Classes	Interfaces
Process Executor.lvclass	IProcess Executor.lvclass
Process Executor Mock.lvclass	ITSDiscovery.lvclass
TSDiscovery.lvclass	
TSDiscovery Mock.lvclass	



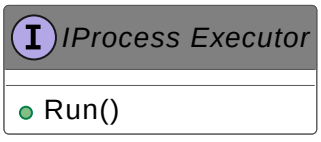
3.2. IProcess Executor.lvclass

Responsibility: IProcessExecutor defines a minimal contract for executing an external process based on an Execution Plan.

The interface exposes a single abstract method, Run, which accepts an execution plan and an error input. Concrete implementations define the actual execution behavior.

Version: 1.0.0.0

3.2.1. Diagram



3.2.2. Methods

Table 11. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Run		Run executes an external process as defined by an Execution Plan. The method starts the specified executable with the provided command-line arguments and applies the configured timeout behavior. If the timeout is set to -1, the method waits indefinitely for the process to terminate; otherwise, it waits up to the specified number of seconds and reports an error if the process does not complete in time.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.2.3. Class Constant VIs

NOTE	No Constant VIs Found

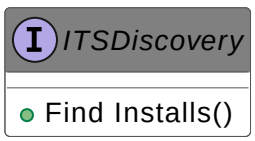
3.3. ITSDiscovery.lvclass

Responsibility: ITSDiscovery defines a minimal contract for discovering installed TestStand versions on a system.

The interface exposes a single abstract method, Find Installs, which returns the detected installations as an array of TSInstall objects.

Version: 1.0.0.0

3.3.1. Diagram



3.3.2. Methods

Table 12. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Find Installs		FindInstalls.vi discovers all NI TestStand installations available on the current system and returns them as an array of TSInstall elements. This abstract implementation only defines inputs and outputs.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.3.3. Class Constant VIs

NOTE	No Constant VIs Found

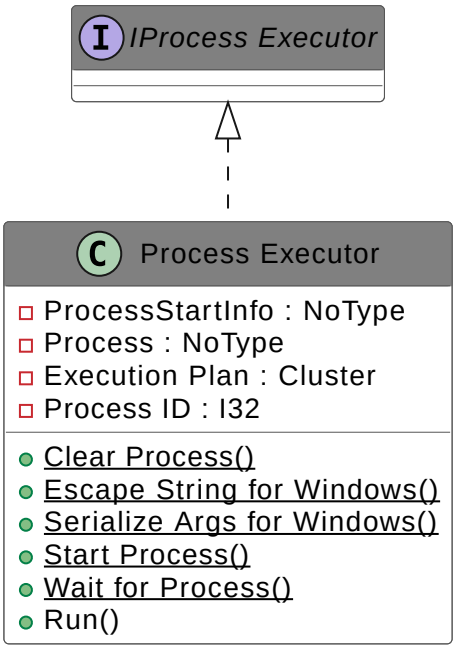
3.4. Process Executor.lvclass

Responsibility: ProcessExecutor executes external processes using the Windows .NET process API.

It translates an Execution Plan into a running system process, handling argument serialization, process startup, and timeout behavior in a consistent and controlled manner. This class contains all process-related side effects and serves as the production execution mechanism for launching tools such as the TestStand Sequence Editor or the TestStand Version Selector.

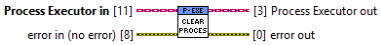
Version: 1.0.0.4

3.4.1. Diagram



3.4.2. Methods

Table 13. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Clear Process		Closes references to a process and its start information object, then clears related data in the process executor object.			
Escape String for Windows		Prepares a single command-line argument for safe use on the Windows command line. The VI escapes double quotes and backslashes according to Windows parsing rules when quoting is required, ensuring that the argument is interpreted correctly by the target process.			
Run		Launches an external executable with arguments and manages its execution timeout.			
Serialize Args for Windows		Converts an array of tokenized command-line arguments into a single Windows-compatible argument string. The VI applies quoting and escaping rules to each argument as needed, ensuring correct handling of spaces, quotes, and backslashes, and then concatenates the serialized tokens in the proper order.			
Start Process		Creates a complete ProcessStartInfo structure from a provided Execution Plan and starts the specified external process accordingly.			
Wait for Process		Waits for a process to complete, optionally enforcing a timeout. If the timeout is >0 the VI waits for the specified seconds for the process to complete. If a timeout occurs, an error is produced. If the timeout is <0, the VI waits indefinitely for the process to complete. If the timeout is 0 the VI exits immediately, letting the process run in the background (fire and forget).			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.4.3. Class Constant VIs

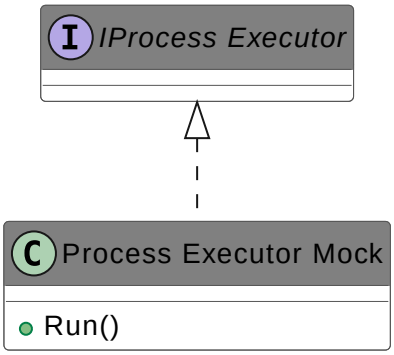
NOTE	No Constant VIs Found

3.5. Process Executor Mock.lvclass

Responsibility: Mock implementation of Process Executor.

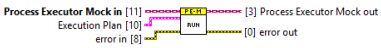
Version: 1.0.0.0

3.5.1. Diagram



3.5.2. Methods

Table 14. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Run		Mocks the Process Executor for testing purposes. In normal execution this VI does not produce an error. If an error is provided as input, it is passed on.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.5.3. Class Constant VIs

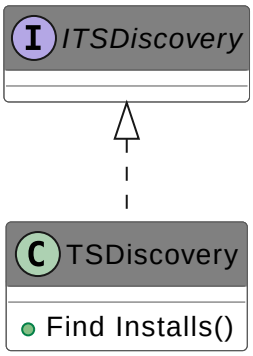
NOTE	No Constant VIs Found

3.6. TSDiscovery.lvclass

Responsibility: TSDiscovery implements the logic to discover TestStand installations on the system.

Version: 1.0.0.0

3.6.1. Diagram



3.6.2. Methods

Table 15. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Find Installs		Discovers all TestStand installations available on the current system and returns them as an array of TSInstall elements. The method is responsible for detecting installed TestStand versions and populating the defined metadata fields (Version, Architecture, Version Key, ID).			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.6.3. Class Constant VIs

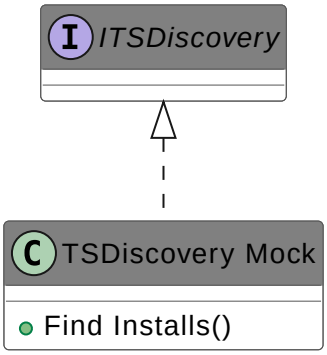
NOTE	No Constant VIs Found

3.7. TSDiscovery Mock.lvclass

Responsibility: Mock implementation of TS Discovery.

Version: 1.0.0.0

3.7.1. Diagram



3.7.2. Methods

Table 16. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
Find Installs		FindInstalls.vi returns a predefined set of TSInstall instances and does not perform any real system discovery. In this mock implementation, the method simulates a system with the following TestStand installations: - TestStand 2025 (64-bit) - TestStand 2025 (32-bit) - TestStand 2023 (64-bit) The returned data is static and fully deterministic, enabling repeatable unit and component tests that are independent of the host system. This implementation is intended solely for testing and validation purposes and must not be used in production code.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.7.3. Class Constant VIs

NOTE	No Constant VIs Found

Chapter 4. Custom errors

TIP Custom errors are added via vi named `*--error.vi`.

Table 17. Custom errors

Name	Code	Description	Owned by
Module Not Running	0		TS Environment Manager.lvlib
Module Not Stopped	403682	The Stop Module VI for the %s module timed out while waiting for the module main VI to stop. The module main VI may still be running.	TS Environment Manager.lvlib
Module Not Synced	403683	%s Module was unable to synchronize events.	TS Environment Manager.lvlib
Request and Wait for Reply Timeout	403686		TS Environment Manager.lvlib
Executable does not exist	510000	The specified executable does not exist.	<<>>
File is not executable	510001	File is not an executable.	<<>>
Timeout during execution	510002	Timeout during execution.	<<>>
Process Exit Code not Zero	510003	Process completed with exit code different than 0: %d	<<>>
Specified version not found	510004	Specified TestStand version not found: %s-%s	<<>>
Specified environment does not exist	510005	The specified environment does not exist	<<>>
Specified file is not a TestStand environment	510006	The specified file is not a TestStand environment (.tsenv)	<<>>
Version not installed (no architecture available)	510007	Can not activate TestStand version %s (no architecture installed)	<<>>
Profile already exists	510008	Profile %s already exists. Delete and recreate to overwrite.	<<>>
Profile not found	510009	Profile %s not found.	<<>>

Name	Code	Description	Owned by
Unrecognized CLI Command	510010	Unrecognized CLI command.	<<>>
Invalid Architecture	510011	Invalid architecture (must be 32 or 64)	<<>>
Invalid Version Format	510012	Invalid Version format (must be Major.Minor)	<<>>
Invalid CLI parameter input	510013	Invalid CLI parameter input. See help for correct syntax.	<<>>
Invalid profile name	510014	Invalid Profile Name (Empty String)	<<>>

Chapter 5. Legal Information

5.1. Document creation

This document has been generated using the following tools.

5.1.1. Antidoc

Project website: Antidoc

Maintainer website: Wovalab

5.1.2. Asciidoc for LabVIEW™

Project website: Asciidoc toolkit

Maintainer website: Wovalab

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

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- 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.
- 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.

5.1.3. Graph Builder

Project website: Graph Builder

BSD 3-Clause License

Copyright © 2020, Cyril GAMBINI All rights reserved.

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

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- 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.
- 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.

5.1.4. classy Diagram Viewer

Project website: classy Diagram Viewer

BSD 3-Clause License

Copyright © 2021, Tatiana Boyé All rights reserved.

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

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- 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.
- 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.

5.2. Product used in the project

The documented project has been developed with the following products.

5.2.1. DQMH®

Copyright © 2021 DQMH® Consortium, LLC. All Rights Reserved.

Find more details on DQMH® Consortium website