MeVisLab Release Notes

Table of Contents

Release Notes 3
- Version 4.1.3 Stable Release (2025-06) 3
  - New Features 3
  - Fixes 3
- Version 4.1.2 Stable Release (2025-05) 3
  - New Features 3
  - Fixes 3
- Version 4.1.1 Stable Release (2025-03) 4
  - New Features 4
  - Fixes 4
- Version 4.1 (2025-01) 5
  - Fixes and Enhancements (MeVisLab) 5
  - Modules and interfaces provided by Fraunhofer MEVIS 10
- Version 4.0.2 Stable Release (2024-09) 11
  - New Features 11
  - Fixes 12
- Version 4.0.1 Stable Release (2024-07) 12
  - New Features 12
  - Fixes 12
- Version 4.0 (2024-05) 13
  - Deprecations 13
  - Fixes and Enhancements (MeVisLab) 16
  - Modules and interfaces provided by Fraunhofer MEVIS 19
- Releases before 4.0 20

Release Notes

This document lists the most relevant changes, additions, and fixes provided by the latest MeVisLab releases.

Version 4.1.3 Stable Release (2025-06)

New Features

- Added the possibility to limit the rendering and interaction of CSOs to a specific viewer identified by its ID.
- Some third-party Python libraries have been updated as usual for a patch release.

Fixes

- Reverted mimalloc third-party library to an earlier version to avoid a memory leak in MeVisLab with the current library version.
- Translation: Fixed an issue where some translation hints could be missing in the created translation files.

Version 4.1.2 Stable Release (2025-05)

New Features

- The `SoView2D` module got a `userInfo` attribute that can be displayed by the `SoView2DAnnotation` module.
- `DicomImport` module: Added the possibility to check the SortPart configuration before applying it.
- Third-party libraries have been updated as usual for a patch release. A notable update is the NumPy Python package, which has been updated from version 1.26.4 to 2.2.6.

Fixes

- Patched Qt to improve startup performance of OpenGL back buffer generation. This affects, e.g., remote rendering of module panels.
- Fixed a problem where active MeVisLab profiling could change the sequence in which fields were updated.
- MATE:
  - Improved Dark mode handling on Windows 11.
  - Fixed **Goto Definition** in the context menu of Python code, which would previously jump to the same location as **Goto Assignment**. This might be slow, though.
  - Fixed a problem where an automatically created type hint import for Python code could be inserted in a statement continued from the previous line.
- Fixed the `Matplotlib` control in MeVisLab when the `hideToolBar` attribute was set.

- Fixed that the Python function `mlabAsync.fieldChanged` did not handle cancelled futures.
- Fixed an initialization problem of the `SoWEMBulgeEditor` module, which prevented it from working properly when freshly instantiated in a network.
- `ImageToCSOVoxelSet` does not print an error anymore when no image is connected.
- Fixed a crash in the `WEMRayInterset` module when connected to a WEM with no patches.
- `DicomImport`:
  - Fixed that 2D images were not correctly split with the **Split time points** option.
  - Fixed that the MeVisLab private frame number tag was missing for single slices from multi-frame images.
- Managed Interaction handling in `SoView2D`:
  - Offset actions did not consider the view the mouse was over when triggered through a key press, e.g., when the slicing operation was bound to key presses.
  - `GenericSoView2DPointingAction`: The `inViewer` field would not correctly update if the corresponding `SoView2D` did not occupy the whole viewing area.
- `PythonPip`:
  - Fixed an exception if Python packages had a `license` directory in their metadata.
  - Greatly improved the startup performance.
- ApplicationBuilder: Improved SBOM generation.

Version 4.1.1 Stable Release (2025-03)

New Features

- MeVisLab-specific Python wrapper classes are now also registered in the virtual package 'mevislab' (previously, they would only be available from '_PythonQt.private').
- Third-party libraries have been updated as usual for a patch release.

Fixes

- A check for still running processes in the Windows installer/uninstaller has been removed again. It did not work for new Windows 11 installations and prevented installation, and also was used in cases where it was not needed.
- The `SoVertexAttribute4ub` module (and perhaps other modules that take integer lists) would not parse value lists that were given as hexadecimal values.
- Python auto-completion for external editors (e.g., PyCharm) showed an bogus extra parameter for some Qt methods. This has been fixed.
- MATE: When creating Python function stubs for MDL commands that take scripting classes as parameters, the type hints for these are now generated as strings, to prevent an import error if these scripting classes are loaded only on demand, e.g., when a panel is opened.

- MATE: Fixed dark mode display of intermediate ReST code for mhelp files.
- `DicomDeidentify`: Changed pseudonymization to return same values as before the 4.1.0 release when all relevant values are the same.
- `SoSelection2` has been un-deprecated, as there currently are use cases that cannot be implemented with `SoPicking`.
- Fixed an exception in the **Interaction Example** of the `MatplotlibMDLExample` module.
- Assorted fixes and improvements for the new `CategoricalEncoding` module.
- Fixed a memory leak in the destructor of the `AccumulateXMarkerLists` module.
- Application Builder: Restrict number of threads used by the installer pre-compression on Windows. We encountered cases where installers could not be built on machines with many cores because the required memory could not be allocated.

Version 4.1 (2025-01)

Fixes and Enhancements (MeVisLab)

IDE

- MeVisLab now has a full-text search for its documentation. The entry for it can be found in the **Help** menu of the menu bar. (Note: The input widget above this entry only searches in section headers.)

  The documentation of user packages can be indexed with the `IndexUserPackagesHTMLFiles` module.
- Input and parameter connections in a network can now be copied instead of just moved to another input connector by pressing the **Shift**+**Control** keys while dragging.
- A newly installed MeVisLab will default to only show the module panels for the currently active network. This behavior can be changed by deselecting the menu item **Panels** => **Hide Panels of Invisible Networks**.
- Support for application translations has been improved, e.g., detection of translatable multi-line strings in Python code, translation hints in comments.
- Since the command-line help of MeVisLab is very long, it is shown in a browser window now instead of a dialog.
- The automatic panel of each module now shows a column that displays - in abbreviated form - the attributes of each field, e.g., is it editable, has it still its default value. Fields can also be sorted by this column.
- File links in the output console get a different color if the linked file does not exist.
- If the variable `DisableModuleWindowsPersistence` is set to `True` in the `mevislab.prefs` file, only the positions of currently open panels are saved in a network file and all others are removed. Previously, all panel positions were removed when this option was set.
- Fixed that **Restart with Current Networks** also opened recent files if the **Auto Load Recent Network Documents** setting was set.

- Fixed that the parsing of floating-point numbers in files depended on the set locale, which could accidentally be set to the system's default. This would affect users with, e.g., German system settings.
- Suppressed a “Unable to open monitor interface” warning from Qt that could occur if the monitor setup was changed on Window.
- Fixed the links created by `:field:` and `:module:` Sphinx roles in network note items.
- Fixed many (mostly small) memory leaks.
- Disabled the broken **Restore Defaults** button in the Preferences dialog for now.
- Lots of small documentation updates and corrections.

MATE Text Editor

- MATE got an option to auto-format Python code with the **black** tool either manually or on each save.
- The Jedi Python package used in MATE for the Python auto- completion has been updated to a recent version, and should support current language features.
- Auto-generated Python function stubs will use the type hint support of Python now.
- The Python debugger will mark object attributes that actually are a 'property' object as such.
- The Python debugger in MATE could show overly long value tooltips. These now have a sensible maximum length, and occasional word breaks for better usability.
- Raise the IDE window when continuing after hitting a breakpoint in the debugger.
- The `MATE` object available for user scripts in the MATE text editor can now also be used in its scripting console.
- MATE: Fixed hard to read text in the workspace file finder drop-down list.
- Fixed the integration of the **rope** Python refactoring library.
- Fixed that the **Alt**+**Down** key combo would not select the next entry in the 'outline' view anymore (while **Alt**+**Up** still worked).
- Fixed a crash in MATE when setting/resetting the busy state cursor with a UserScript.
- Fixed a crash if **Ctrl**+**F3** is pressed without selecting or searching something before. Now, the text under the cursor is automatically selected in this case.
- Fixed that the **Close all secondary windows with ESC** preferences option inexplicably also was used for opening the error check view of Python code. This has now its own option.

TestCenter

- The `ScreenShot.createScreenShot` function in the `TestSupport` Python package now allows to specify that the screenshot should not be grabbed from the screen, but instead be done in an offscreen buffer, which works around the problems of panels sometimes being obscured by other windows or the screen saver.
- Added missing compare functions to `TestSupport.Base`
- Fixed that a test could sometimes be started recursively a second time by the user while it was still running.

Scripting

- Added the function `MLAB.isNone()`, which can be used to check if a wrapped Qt object has been deleted. This method has been added because the usual way of simply using the object itself as bool value could return wrong values because it would also use any `isVisible`, `isNull`, or `isValid` method of an object.
- Added the function `MLABDicom.loadDicomTreeFromBinaryContent`, which loads a DICOM file from a buffer instead of from a file.
- Users can unset the internal "should stop" flag with `MLAB.setShouldStop(False)`. This can be helpful in server scenarios.
- Named dock presets/layouts of the IDE can be activated by scripting with `MLAB.IDE().activateDockPreset(name)`.
- `CurveList` and `CurveData` objects can now be created by scripting with `MLAB.createMLBaseObject("CurveList")` and `MLAB.createMLBaseObject("CurveData")`, respectively.
- Some code wizards were adapted to not create Python code with (initially) unused imports.
- Jobs submitted to the `JobScheduler` extension were never finished if the executable to use did not exist.
- `JobScheduler`: Fixed that jobs that were chained to a predecessor job that didn't exist anymore were not necessarily removed.
- Fixed that adding a Python function as field listener to a module with no own script context would fail.
- Fixed `CSOWrapper.getIsPointInside` behavior.
- Made sure that the output from the `logging` default logger is correctly categorized in the output console as info/warning/error.

Controls

- The `ProgressBar` control now shows an indefinite progress bar when combined with a boolean field. One needs to call `MLAB.processEvents(True)` periodically to allow GUI updates of the progress bar, though.
- Fixed that the CSS `hover` attribute was ignored for buttons associated with a field when using CSS styling for a panel GUI. This was only a problem when used in the IDE, though, it always worked when run as application.
- Alleviated a problem where the GraphicsView would not show the content of a web item if the view is too big. The issue still exists, but the view can be larger now.

C++

- For Visual Studio projects, UTF-8 is now the expected encoding for source files.
- Windows: By default, the `/MP` option is used when compiling C++ modules with Visual Studio; this can be configured with the `MLAB_MSVC_PARALLEL` CMake option.

Application Builder

(for those users with an ADK or Application Builder license)

- Installers can now generate a rudimentary JSON software bill of materials (SBOM) file. Set the variable `GENERATE_SBOM` to `1` in your `.mlinstall` file to use this. The SBOM file is generated alongside the installer.
- Fixed annoying misleading warning message about files that could not be decrypted (and were not meant to, because they were provided this way by the MeVisLab SDK).
- Fixed: Installing Python packages with `PythonPip` generated unnecessary (optional) library dependencies, which could mean unnecessary Python libraries were included in standalone installers if they were also installed.
- Fixed a crash in the **Standalone Installer** wizard when closing the wizard after typing an invalid or incomplete application module name.
- Windows specific
  - Installer generation now uses **7-zip** for pre-compression, which yields far better compression, but takes somewhat longer.
  - Warn if an application can not be uninstalled completely because some executable from it is still running.
  - Fixed that installers installed by users without admin rights would sometimes not work because the required updated VisualC runtime could not be installed on the system. Now, the necessary DLLs are installed into the application directory in this case.
  - Fixed uninstall of long file paths.
- Linux specific
  - Now allowing to select an executable (process) name for the standalone installer that is identical to the application name.
  - Allow installers with spaces in their product name.

Third-Party Libraries

A lot of third-party libraries have been updated in this release.

See the list of available third-party libraries here.

Note that this is not the complete list of third-party components used in MeVisLab; for this, you should consult the About dialog of MeVisLab, or use the `ThirdPartyInformation` module.

Some notable updates/additions:

- mimalloc is now used by MeVisLab to improve memory allocation efficiency on Windows (only in Release mode).

Modules

- `DicomImport`:
  - The user can now drag directories or files to import onto the module.
  - Added new modules for filtering which files are imported: `DicomImportTagFilter` and `DicomImportBooleanOperationFilter`.

  - Added JPEG2000 decoding support.
  - Allow slightly wrong DICOM files with trailing zero bytes instead of trailing spaces for odd-sized string tags.
  - Added support for segmented palette LUTs.
  - `DicomImport` now prevents import of more than one file with the same SOPInstanceUID.
  - Fixed: Tags stored in SharedFunctionalGroupsSequence tags were ignored during import of multi-frame files.
- `SoView2D` and `SoDiagram2D` by default now use the Qt library for 2D text rendering, which has the advantage that it usually can print all unicode characters. The old default is restored by setting `View2DEnableQtFontRendering = No` in the MeVisLab preferences file.
- `PythonPip` will not propose a target package for installing packages by default, the user must select one (but the selection will be remembered).
- The `GraphManager`: module got the ability to create a completely empty `Graph` object (which can be filled from scripting).
- Improved loading speed of `Graph` objects with the `BaseLoad` module.
- `SoCSOTransform` modules will now accumulate transformations in typical OpenInventor manner (only for newly created modules that use this information - existing modules in saved networks will retain the old behavior through a hidden option field).
- Object inputs and outputs of ITK wrapper modules are now more specifically typed, which should make it easier to connect the correct input objects.
- The `itkImageFileReader` and `itkImageFileWriter` modules have been moved to the `MeVisLab/ITK` package.
- Changed the `Histogram` module to never set a NULL object at its output.
- `ImageLoad`: Make PNG loading more robust against unusual compression parameters.
- Fixed that the progress bar in `WEMLevelSet...` modules didn't show any intermediate progress.
- Fixed: Some `WEMLevelSet...` modules could crash if applied on a `WEM` without any faces.
- `UndoManager`: Fixed cause of rare, spurious crashes on module creation.
- Fixed a crash in the `MemoryCache` module if the input image is too large. Instead print an error message.
- `VideoWriterML` and `VideoWriterInventor`: Fixed handling of non-ASCII characters in the target filename.
- `ExportImageAsSlices`: Fixed that the **Export** button was disabled when a valid file path was already set upon network loading.
- `Thumbnail`: Fixed auto-update behavior.
- The `ExtractRTStruct` module did not produce any output if the SourcePixelPlanesCharacteristicsSequence tag was not empty.
- Fixed image border handling of `View2DIsoContourShader`.

- `So3DMarkerRenderer`: Fixed that the internal bounding box of this scene node did never shrink when markers were removed.
- The `SoView2DMarkerEditor` has been refactored somewhat.
- `WEMIsoSurface`: Will now unset its `WEM` output in `AutoUpdate` mode if the input image is removed.
- Fixed: `WEMGeodesicClip` was only usable in AutoUpdate mode.
- `WEMBinarySurface`: Fixed that profiling didn't work for this module.
- Fixed: Connecting an invalid image to a `CSOIsoGenerator` cross-connected to a `CSOMerge` triggered an endless loop.
- Fixed a problem with the `SoCSOEllipseEditor` that path points would not be correctly calculated while dragging the cursor outside the viewer.
- `SoView2D`: Fixed: It has been possible to set the maximum z, t, and u index for the displayed image beyond the actual dimensions of the input image.
- Fix `SoView2DAnnotation` occasionally printing errors to the console when showing long, abbreviated UTF-8 strings.
- `SoLUTEditor`: Changed the behavior of the interactive editor: Previously, it was too easy to accidentally insert new control points when wanting to move the first or last LUT point up or down. Now a more conscious move to right or left is needed.
- `Sobel3D`: Fixed a potential division-by-zero crash.
- Fixed that the `enabled` field of the `SoPathTracerMaterial` or `SoPathTracerGradientVolume` modules was ignored.
- Deprecations:
  - Removed (deprecated) `SegContourList` inputs from modules `DynaCurve` and `FuzzyConnectDistance`.
  - `FuzzyConnectDistance` has been deprecated, as its proper use has been lost to history.
  - `ImageROIIsFileCache` has been deprecated, use `AccumulateImage` instead.
  - The `StringListSplitter` module has been deprecated.

Modules and interfaces provided by Fraunhofer MEVIS

New modules

- `ConnectedComponentsMacro`, for simple access to the most common features from combinations of `ComputeConnectedComponents`, `FilterConnectedComponents`, and `ConnectedComponentsToImage`.
- `ExtractTextTableFromPRData`, which tries to convert textual data from the `GraphicAnnotationSequence` of DICOM files as plain text, JSON, and CSV.
- `FieldDifferenceViewer`, for showing differences between two strings as a difference view.
- `SelectTimepoints`, which allows to select a subset of time points from an image, either by index, or by which are closest to a list of desired time stamps.

- `MarkerToImage`, for changing image values at provided marker positions.
- `CategoricalEncoding`, which converts one- or multi-hot-encodings to integer labels and vice versa.
- `NormalizeImageByPercentileMapping`, which normalizes image values by linearly mapping lower and upper percentile to a defined range.

Improvements

- Fixes of several code smells and memory leaks; more compile warnings are handled as errors now.
- `SoMLMatrixTransform` can now invert the matrix on the fly.
- DICOM related modules:
  - Many DICOM documents updated.
  - `DirectDicomImport`: Also uses the new JPEG2000 decoder mentioned above now (previously it used the `itkImageFileReader` for JPEG2000 encoded image data).
  - `DirectDicomImport`: Does not compose certain IODs / Modalities anymore, which it previously did in error.
  - Improved robustness against non-standard and broken DICOM files, improved issue logging.
  - Added some checks and warnings for error cases that might be caused by non-standard, or too long, file names.
  - `DicomTreeCompare` has new parameters to control difference dumps.

Bugfixes

- `CSOBoolOp`:
  - Fixed a crash that occurred when the input `CSOLists` contained identical `CSOs`
  - Fixed incorrectly empty results when the input `CSOLists` had two path points in different `CSOs` that were very close together (< 1e-9 mm)

Removals

- Removed the project `MLStochasticCollocation` (`...LinearFunction` and `...InterpolationPolynomial` modules) from the `FMEwork/Release` package, as nobody seemed to use it.
- The `PCLMarchingCubesRBF` module been removed since the used PCL algorithm is too memory- and time consuming for nearly all parameter configurations, so that is was not stable enough for normal operation.

Version 4.0.2 Stable Release (2024-09)

New Features

- Qt has been updated from version 6.6.3 to 6.7.2. Other assorted third-party packages have also been updated.

Fixes

- Fixed pixel spacing and patient orientation for single images imported with `DicomImport`, when the corresponding tags are contained in a `SharedFunctionalGroupSequence` tag.
- Fixed that `DicomImport` would sometimes swallow the image data of files when imported together with other files (in the same series) not containing image data.
- Fixed: `DicomImport` would set incorrect image min/max values for files where these values were not explicitly given but had to be derived from the number of bits used.
- Fixed: `DicomImport` did not correctly handle PALETTE COLOR images. This led to garbled images or crashes on accessing the output image.
- Fixed an occasional crash of remote modules using the `RemotePanelRendering` module for remote UI.
- Clean-up and fixes of license identifiers displayed for some third-party libraries.
- Fixed that the context help function in the MATE editor (e.g., for scripting functions) wouldn't jump to the relevant position in the help file.
- Fixed: The OutputInspector would show the first inspected output twice if the view is in a floating window (instead of docked to the main window).
- Fixed: Editing network notes did not allow to add links by shift-clicking GUI elements anymore.
- Fixed: Some mouse interactions on a network didn't work on displays with a configured scaling other than 100%.
- Fixed: The ML Module wizard had created invalid code.
- Fixed some duplicated WEM help entries when using the **Help** → **Search in MeVisLab** function.
- Removed some help sections pertaining to the discontinued macOS support.
- ADK/ApplicationBuilder: Removed non-functional macOS settings from the Standalone Application wizard dialog.
- ADK/ApplicationBuilder: Fixed stand-alone applications not working correctly if they use the asyncio event loop, but do not (even indirectly) import the Qt Python bindings.

Version 4.0.1 Stable Release (2024-07)

New Features

- Various small updates for third-party libraries. A notable change is boost, which was updated to version 1.85.0 (from 1.84).
- The `PythonPip` module will now try to determine a SPDX compatible license identifier for newly installed Python packages when generating the `.mlinfo` entry for the **About...** box.

Fixes

- Windows: The SDK installer didn't install the Visual C runtime installer, so it failed if the runtime wasn't installed by some other application first.

- Fixed: `DicomRescale` could crash if operating on a DICOM file with invalid tags.
- Fixed various other crashes caused by loading invalid or corrupt DICOM or .mlimage files.
- Fixed loading of DICOM tags which have more than one allowed VR type.
- Updated the MESA OpenGL driver to fix a crash triggered by the `SoPostEffectAntiAliasing` module when using software rendering on Windows.
- Fixed a crash that could occur in the `ListView` control when adding more columns (via scripting) to items containing rich text.
- Fixed a crash in secure mode testing when opening panels with stored relative positions.
- Fixed unvoluntary pinning of wheel events in the `RemotePanelRendering` module.
- Fixed: The `BoundingBox` module didn't crop the C/T/U dimension information to the output dimensions (unlike, e.g., the `SubImage` module).
- MATE: Fixed hardly readable text in workspace file finder.
- Small documentation fixes.

Version 4.0 (2024-05)

Deprecations

We mainly used this release to remove old interfaces and modules that were deprecated, or to fix interfaces that had severe drawbacks.

Modules

The following deprecated modules have been removed from MeVisLab (you could instantiate them in previous versions if you knew their exact name, or if they were used in old networks).

- SoAsciiText
- CSOBulgeEditor, CSOFreehandProcessor, CSOIsoProcessor, CSOLiveWireProcessor, CSOModifyProcessor, CSOPrimitiveProcessor, CSOTransformationProcessor, SoView2DCSOEditor
- SoFixedFunctionShader
- SoAngleLines, SoAngleToObjects, SoCake, SoDistanceLine, SoMainAxis, SoMinimalDistance, SoRuler, SoShapeToPointSet, SoThresholdToPointSet,
- SoScreenSpaceAmbientOcclusion
- ImportDialog, OISettings, OpenImage, SaveImage
- EatDicomImport, DicomService
- Negation
- AngleLines2D3D, DistanceLine2D3D, SimpleImageStatistics

- ObjMgrEventClient (and all ObjectManager related modules)
- LiveWireMacro
- SwapViewer
- BaseBypassOp
- ConnectedComponents
- ContourManager, Draw2D, LiveWire, RegionToContour
- DistanceTransform
- LoG
- KernelFilter
- WEMClip
- MovieCreator

Python

- The following MeVisLab-specific Python packages were deprecated and have been removed:
  - `ObjMgr` (as the associated modules have been removed, too
  - `matrix4` (use `linear.matrix4`)
- The following MeVisLab-specific Python functions and methods were deprecated and have been removed:
  - `linear.vec3.copy` (use `vec3` constructor instead)
  - `linear.toVec3` (use `vec3` constructor instead)

QtWebKit

The old QtWebKit-based web control has been removed in favor of the (still supported) QtWebEngine based one. This new web control supports modern web standards and is based on the Chromium engine (used by the Chrome browser), but has one important drawback: Since the HTML/JavaScript engine runs in a separate process, communication with MeVisLab needs to be serialized, and viewer controls can't be embedded into an HTML page, at least not directly.

The `TestWebEngineView` module demonstrates the abilities of the `WebEngineView` control.

Note

Technical note: Since the `QWebEngineView` rendering in Qt6 is based on `QtQuick`, we had to switch the QtQuick rendering backend to "OpenGL" to enable co-existence with the OpenGL/OpenInventor viewers in MeVisLab.

If you implement your own QtQuick based widgets in MeVisLab, we suggest to set the backend to "OpenGL" in your code, too, with the following line:

```
QQuickWindow::setGraphicsApi(QSGRendererInterface::OpenGL);
```

to avoid render behavior changing depending on the module load order.

C++

- Lots of old names of renamed functions and enums have been removed.

  There still exists a Python script file to help in the renamings: `Packages/MeVis/BuildSystem/BuildTools/Scripts/replaceDeprecated.py`

  You can use this script from MeVisLab with the `MLReplaceDeprecatedAPI` module.

- Some methods were replaced because they returned a const char* object, where it was the recipient's duty to free the allocated memory afterwards. In these cases there are methods that take a std::string instead.

- Some methods were removed because the task could nowadays as easily be performed with methods from the STL (or Qt).

- The method `ml::Module::calculateOutputImageProperties` now has two arguments. The second one is the `PagedImage` on which to set the image properties (previously obtained with `getOutputImage(outIndex)`).

  You might notice that trying to override the method with one argument now causes an error in the compiler. This is intentional, you need to override the method with two arguments.

- `getFieldContainer()` isn't needed anymore, as `ml::Module` is derived from `ml::FieldContainer` anyway.

And so on.

If you are unsure how to replace removed methods, install the 3.7.2 SDK, and compile your code with deprecation warnings enabled, and/or look at the documentation of the methods.

Qt6

Qt has been updated to Qt 6.6. Consequently, some long deprecated classes and methods have been removed. Some of the more prominent ones are:

- `QGLWidget` is no more. Use the newer `QOpenGLWidget`.

- `QRegExp` has been superseded by `QRegularExpression`.

- `QTextCodec`, `QTextEncoder`, and `QTextDecoder` have been removed (not exactly, they still exist in the QtCore5Compat package, which we don't include in the SDK). By default, Qt only supports Latin1 and UTF encoding.

- The multimedia framework has undergone some refactoring.

Deprecated Modules

The following modules have been deprecated in MeVisLab.

- MessageBox

- MessageBoxExt

- OverwriteVectorComponent

Fixes and Enhancements (MeVisLab)

IDE

- The default encoding of text files read or written by MeVisLab is now UTF-8, if not specified otherwise.
- The window positions of modules in a network are now stored relative to the screen the IDE main window is on.
- Added a new command line argument `-prefs-variable` to define/override a preferences variable (e.g., 'Locale=en_US').
- Auto-completion for imports in the Python scripting console didn't work for namespace packages (PEP 420).
- When searching for words in an editable text area with the search dialog, the highlighted words were hardly visible.
- MATE: Fixed shortcut **F7** being defined twice in the help editor.
- When converting a local to a global macro, a dummy example network is now created automatically.
- **Ctrl**+**L** doesn't just switch to the **No Dock Windows** layout now, but toggles between this layout and the previous one.
- When changing the visibility of an connector by removing the hidden tag again, a reload of the module would not update the hidden state.
- Fixed that a field comment from the Interface section is lost if the field is also listed in the Description section.
- MATE: The project workspace search now supports multi-word search.
- The MeVisLabStarter executable (on Windows) doesn't have any special functionality anymore, and is just needed to restart MeVisLab. It sets the -singleinstance command line parameter of MeVisLab though, so if you want to re-use a running instance of MeVisLab, you can continue using MeVisLabStarter or set the -singleinstance parameter yourself.
- When opening a scripting console for the field of a module (through the context menu), it would not stay on top like other windows.
- Fixed a crash on closing the IDE when the cache size in the GUI is updated one last time.
- Fixed background color of screenshots of panels that included OpenInventor viewers.
- The **About** dialog now has a link to the license text of MeVisLab.

TestCenter

- The `CodeTest` module in the `TestSupport` package got an optional parameter for its `inject` function that allows to ignore output lines matching a regular expression before doing error detection.
- Fixed **Show Enclosing Folder** context menu entry for wrapped CodeTests.
- Python coverage ignores code in a `site-packages` directory by default (since it contains third-party code). Also added the ability to specify explicit inclusion of directories through the TestCenter config file ("InclusionExpression").
- Images in test results are restricted in the displayed size (but can be expanded).

- Avoid temporarily opening a console window on test cleanup (only happened on Windows).
- If the TestCenter is running in secure mode on Linux, and a test crashes (with a segmentation fault), the TestCenter will try to give a stack trace (which might include the signal handler functions as top-level entries).

Scripting

- The generation of ThirdPartyInformation (About dialog) entries and dependency information for Python packages installed with `PythonPip` got more robust.
- MeVisLab now comes with automatically generated .pyi files for the scripting wrappers available in MeVisLab. These should automatically be found by any Python IDE that uses our Python environment (through `sitecustomize.py`) and help with auto-completion and type checking. If you don't want this, set the environment variable `MLAB_DISABLE_SITECUSTOMIZE` to 1.
- Fixed a crash on destruction of the `SoViewportRegionWrapper`.
- Added the option to not create skeletons in `MLGraphWrapper.createEdge`.
- `MLABScriptProcess` doesn't inherit file handles anymore by default.
- Slightly changed the behavior of `MLAB.setVariable()`, which previously wouldn't change a value that was stored persistently.
- The interface of the class returned by `MLABGraphicsScene.addWebView` has changed somewhat, because there is no native support for adding a web viewer to Qt's `QGraphicsScene` anymore.

Controls

- Fixed layout of `HyperLabel` control with longer texts.
- The `Image` control re-renders SVG images to the displayed size now.
- Keyboard navigation/selection did not work for a `RadioButtonGroup` if it was attached to a field.
- Button titles of `ToolButtonGroups` connected to enum fields were not translated.
- The `PythonTextView` was buggy with regard to the syntax highlighting, which could lead to overlapping line numbers and crashes.

C++

- `ml::CurveList` and `ml::CurveData` are now derived from `ml::RefCountedBase`, which means that these classes should not be deleted explicitly by the user, but implicitly through the use of the `CurveListPtr` and `CurveDataPtr` classes. This also means that CurveList and CurveData can't be direct members of another class anymore.

  Especially the interface of `ml::CurveList` has been adapted accordingly, with explicit methods for adding and removing of `ml::CurveData` objects.

Application Builder

(for those users with an ADK or Application Builder license)

- When local macros share the same name, only the dependencies of one of these modules would be considered.

- Determining DLL dependencies for stand-alone installers on Windows doesn't require an external tool like `DependencyWalker` anymore (and also became a lot faster).

Windows specific

- MeVisLab 4.0 was built with and requires Visual Studio 2022 now.

Third-Party Libraries

A lot of third-party libraries have been updated in this release.

See the list of available third-party libraries here.

Note that this is not the complete list of third-party components used in MeVisLab, for this you should consult the About dialog of MeVisLab, or use the `ThirdPartyInformation` module.

Some notable updates:

- Qt has been updated to version 6.6.3 (from 5.15).
- The PythonQt scripting wrappers for Qt have been re-created.
- Python has been updated to version 3.11.9 (from 3.9).
- jemalloc isn't used by MeVisLab and not included in the SDK anymore.
- MeVisLab now solely uses libjpeg-turbo for the JPEG file format. libjpeg is no longer included in the SDK.
- CMake target names of some third-party libraries have changed (mostly upper/lower case).
- The Python package asyncqt has been replaced by qasync to fix problems with aiohttp.
- ITK has been updated to version 5.3.0, and the module wrappers have been updated, including newly introduced classes.
- VTK has been updated to version 9.3.0. Almost all module wrappers have been removed, please use the Python bindings provided by VTK instead. Have a look at the example networks of the remaining bridge modules for how interfacing with MeVisLab modules works.
- The Windows version now uses 7-Zip in its installer.

Modules

- `DicomImport`:
  - Do not split image series just because they erroneously have slightly differing StudyDate/StudyTime tags.
  - Make DicomImport (and other modules) more tolerant against certain errors in the DICOM data.
  - Do not show corrupt image data for .dcm/.tif pairs. Correctly report missing image data (.dcm/.tif pairs must be read with `ImageLoad`).
- We added an example `DicomViewer` application module to MeVisLab, which provides basic DICOM image viewing capabilities in 2D and 3D.
- The `clear` button in the `WEMInteract` module now clears the drawn contour.

- `WEMModify` updates the face normals now. This attribute is, e.g., used by `WEMInfo` to compute the volume of a WEM patch.
- `SignedEuclideanDistanceTransform`: Fixed wrong output for example network if `minValue` and `maxValue` are set to 0.
- `SoCoordinateSystem` rendered the text shadow in front of the actual text under certain circumstances.
- `CSOMerge` did handle input notifications when auto update was set to off.
- Most WEM modules that use PVLs (primitive value lists): Introduced checks for correct type (per-node, per-edge, or per-face).
- `WEMLoad`: Introduced a new field for auto-loading of WEMs.
- `ConnectedComponentsToImage`: Did set an invalid image extent if no cluster was selected.
- `WEMVascularSystem`: Would hang indefinitely if the graph was completely outside the image extent associated with it (which could happen if you constructed the graph by scripting).

  There is also now a field `clipping` which determines if the resulting WEM should be clipped to the associated image extent.
- Added new color modes to `So3DMarkerRenderer` that use vector length/angle and use a LUT as color input.
- `WEMLoad` now can load .ply files with one-based indexing (previously it crashed).
- `ImageStatistics`: Fixed that the variance could get slightly negative due to rounding errors.
- Fixed rare crash in `SoCoordinateSystem` if no axis was initialized.
- The `OutputInspector` for images now allows to step through the user dimension with **Ctrl**+**Left** and **Ctrl**+**Right**.
- Fixed: The code generated by the Project Wizard for modules derived from `SoView2DExtension` missed an important dependency.

Modules and interfaces provided by Fraunhofer MEVIS

Additions

- Module `DicomPRSave` saves `XMarkerList`, `StylePalette`, and `CSOLists` Base objects as and in DICOM files of `Grayscale Softcopy Presentation State IOD`. See also `MultiFileVolumeListPROutput`.
- Module `FileEncrypter` encrypts files using a new C++ API `BinaryEncryption`, but the currently implemented encryption method is internal to FME, so this module is only useful if you implement your own.
- Module `SoMLMatrixTransform` wraps `SoMatrixTransform` to enable direct use of ML and numpy matrices (which are transposed relative to the matrices used by SoMatrixTransform).

Improvements

- `DirectDicomImport` resolves variables (such as `$(NETWORK)`, `$(TMPDIR)` etc.) in fields **Import Files** and **Import Directories** in the **Configuration** panel.

- Improved robustness of experimental `MultiFileVolumeListPROutput`.
- `LoadYAML` and `ReadYAML` can now deal with Python tuples encoded with "!!python/tuple".
- Macro module `CachedPath` and Python module `file_caching` now support pruning of the synchronization target folder, which is enabled by default.
- Removed the unused/unsupported `NoCache` mode from the modules `ImageCache` and `ImageCacheForNetworkLoops`.
- `ThirdPartyLicenseReport`: now reads 'homepage' as alternative to the 'url' key in .mlinfo files; and some other minor improvements.
- The `fmeTestSupport` Python module now supports `EXPECT_ERRORS()`.

Changes

- The C++ Module base class `XMarkerListProcessor` now allows to specify a number of input images (for modules that combine image and marker data)

Releases before 4.0

For the release notes of older MeVisLab releases see this document.