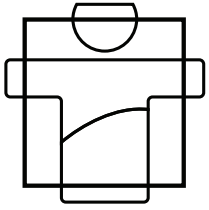
MeVisLab 2.6 OS X Edition

Felix Ritter



Fraunhofer
MEVIS

Contents

Introduction

This document introduces specific features and modifications of the OS X edition of MeVisLab. Please refer to the general MeVisLab documentation for all other aspects not mentioned here.

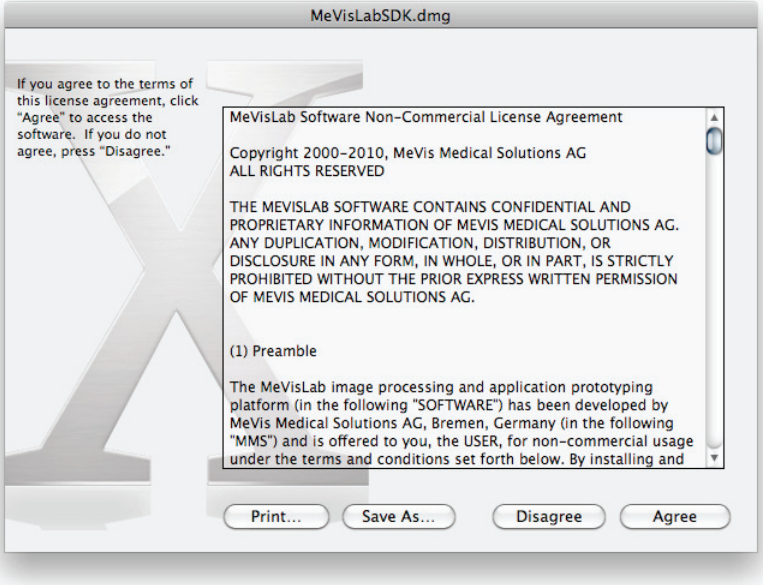
Installation

MeVisLab 2.6 for OS X requires OS X 10.8 (Mountain Lion) and must be run on a 64-bit capable Intel Mac.

Currently, neither the installer nor the application is signed. The Gatekeeper feature of OS X may prevent the installation or execution of MeVisLab depending on its configuration. Read http://support.apple.com/kb/HT5290 to learn more about Gatekeeper and how to exempt MeVisLab from Gatekeeper.

MeVisLab is distributed as a OS X disk image (dmg). Before opening the disk image, the license agreement is displayed. Please read the license agreement carefully before accepting it (Fig. 1).

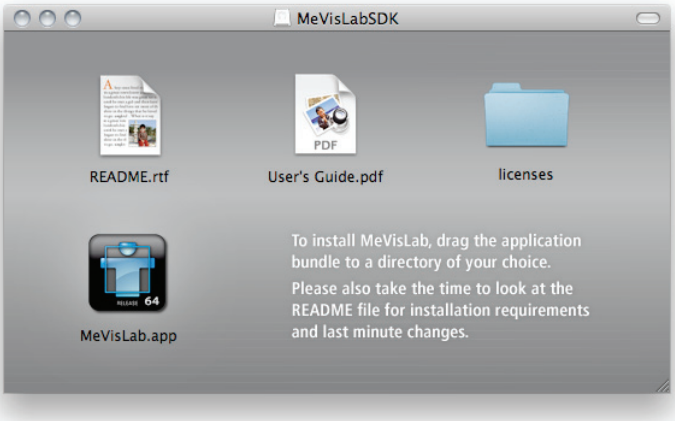
Figure 1 License agreement



Besides the MeVisLab application and full Software Development Kit (SDK), the distribution contains the MeVisLab User's Guide of the OS X Edition and the license document.

The distribution also includes a `README` file with up-to-date installation instructions and requirements, please review both before installing and starting MeVisLab (Fig. 2).

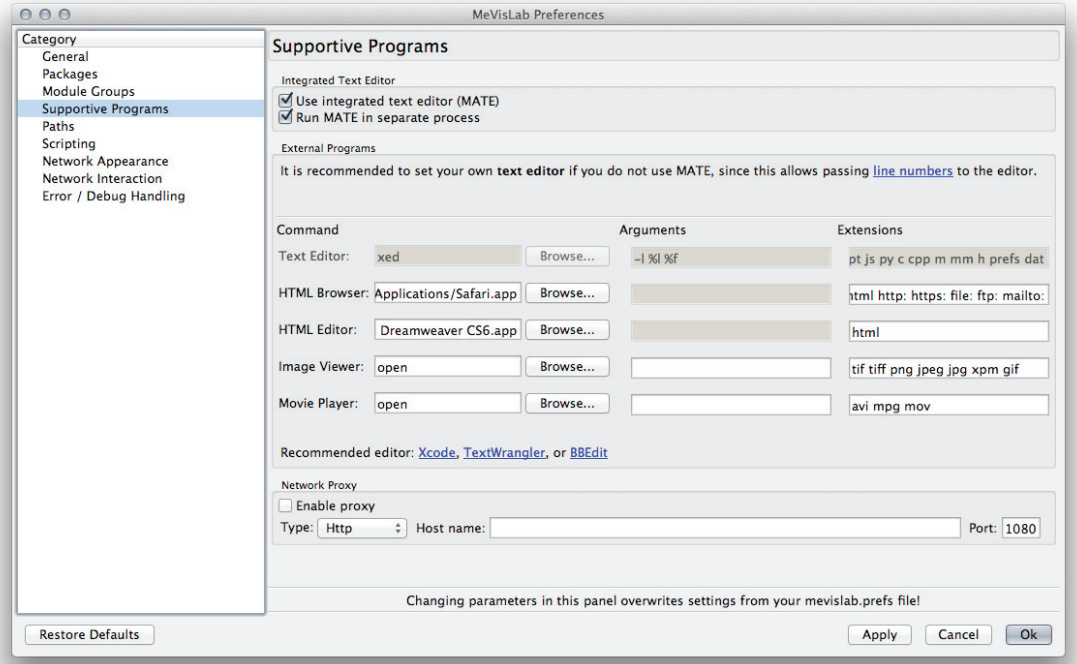
Figure 2 Contents of the distribution disk image displayed in Finder



MeVisLab is built as a self-contained application bundle and may be installed in a directory of your choice. The recommended installation location, however, is within the user or system-wide Applications folder (`~/Applications` or `/Applications`, where '`~`' is your home directory).

You may now proceed to set up the preferences for external default applications, such as your favorite text editor or web browser. This is done within MeVisLab via the Preferences panel (available via menu *MeVisLab › Preferences… › Supportive Programs*). To use the OS X default application for a document type, specify the command `open` (Fig. 3).

Figure 3 MeVisLab Preferences panel



Congratulations, you may now start using MeVisLab. If you would like to extend the functionality of MeVisLab by developing your own MeVisLab modules, please proceed to *Module Development* on page 18.

Mac-Specific Features of the IDE

Some small enhancements have been made to the MeVisLab Integrated Development Environment (IDE), which are only available in the OS X edition.

Error badge on the MeVisLab icon in the dock

Errors in MeVisLab or one of its loaded modules are printed to the MeVisLab Debug Console. In addition to that, the MeVisLab OS X edition also overlays the MeVisLab icon in the dock with an error badge (Fig. 4).

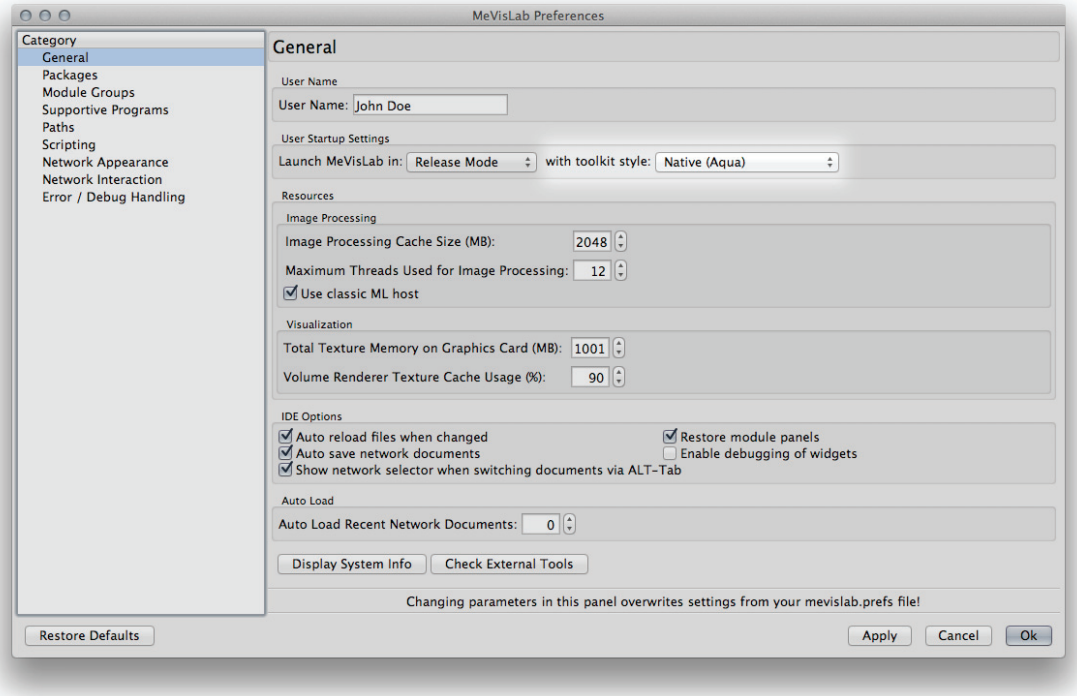
Figure 4 Error badge on the MeVisLab icon

To clear the error state and to remove the badge from the icon, right click in the MeVisLab Debug Console and select *Clear* from the context menu.

Changing the Toolkit Style of MeVisLab

MeVisLab is provided for all major platforms. To facilitate the development of cross-platform modules and MeVisLab applications, you may change the Toolkit Style used by the IDE. To do so choose one of the available styles in the *Toolkit Style* section of the Preferences Panel (*MeVisLab › Preferences... › Panel Appearance › Toolkit Style*, see Fig. 5). This setting requires a restart of MeVisLab to take effect.

Figure 5 Choosing a Toolkit Style for MeVisLab



The Toolkit Style of MeVisLab can be changed temporarily by pressing the Shift key during startup of MeVisLab. MeVisLab will provide audio feedback after a pressed key modifier has been recognized.

A specific style may also be passed via the `-style <style>` command line argument to the MeVisLab starter executable. Valid choices for `<style>` are `Native` or `Application`, e.g.

```
.../MeVisLab.app/Contents/MacOS/MeVisLab -style
Native
```

to use the native OS X style, and

```
…/MeVisLab.app/Contents/MacOS/MeVisLab -style
Application
```

to apply the application style to MeVisLab.

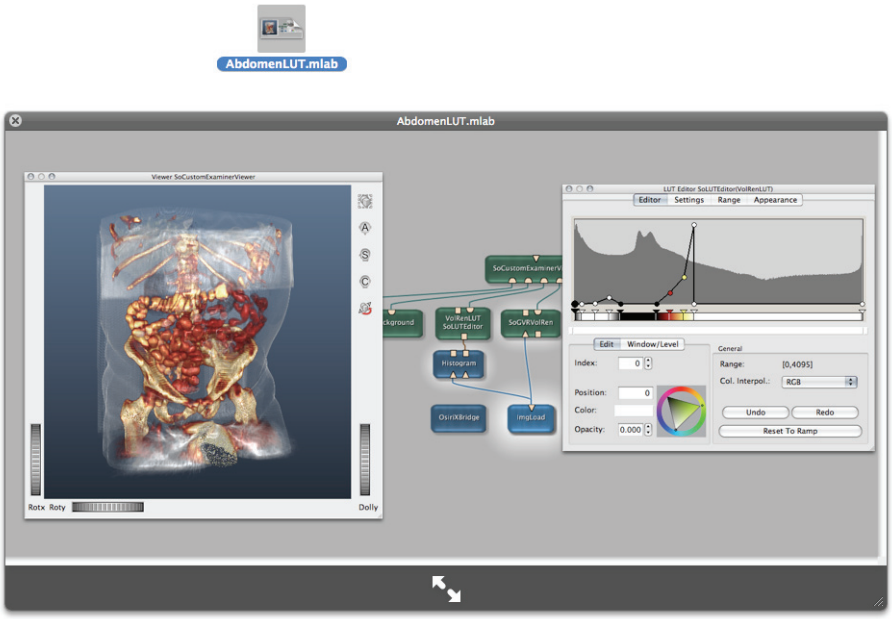
Spotlight importer plugin

MeVisLab associates itself with several file types, such as Scripting documents (.script) and Definition documents (.def). To help you organize, MeVisLab contains a Spotlight metadata importer plugin for its document types. In collaboration with the Spotlight importer provided by Xcode, you may search all files of importance to MeVisLab, except MeVisLab Network documents (.mlab).

Quick Look integration

For a quick inspection, MeVisLab users may exploit the Quick Look functionality of OS X to inspect Network (.mlab), Scripting (.script), Definition (.def), and License (.dat) documents (Fig. 6).

Figure 6 Take a Quick Look of MeVisLab networks on OS X



The Quick Look previews of MeVisLab Network documents are stored in the resource fork of the Network files and thus do not affect the loading of Network documents in any form. Only Network Documents that have been saved with the MeVisLab OS X edition running on OS X 10.5 or better provide a preview image. Besides Quick Look, the previews will also show up in Finder or on the Desktop if you choose so.

OS X Image IO integration

MeVisLab contains an OS X-specific image format plugin that interfaces with the OS X Image IO framework and provides read access to most of the OS X image formats, e.g. using the ImageLoad module. At the time of writing (OS X 10.5.4), supported formats include:

JPEG-2000, Adobe Photoshop, Camera-RAW, GIF, Radiance, Quicktime Image, DNG, PICT, Targa, SGI Image, Windows Icon, Mac Icon

Currently, the following extensions are registered by the plugin:

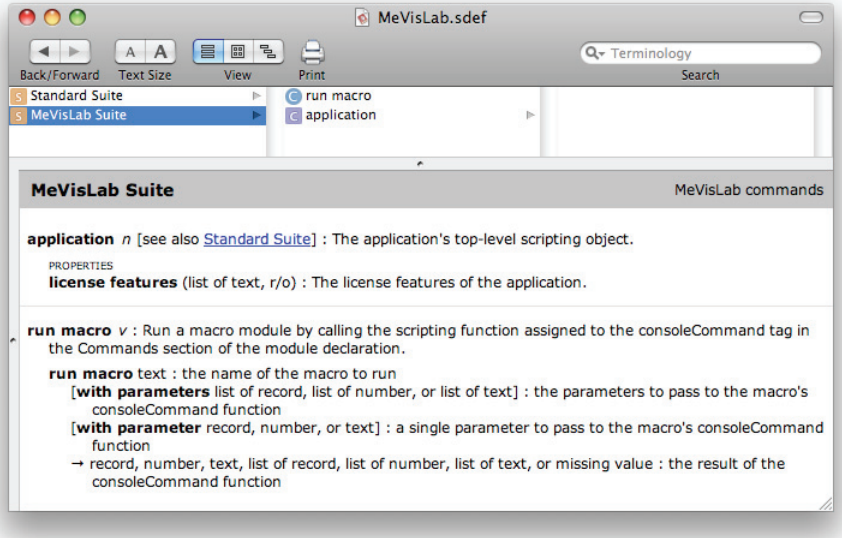
3FR AI ARW BMP CR2 CRW DCR DNG EXR FFF FPX GIF HDR ICNS ICO J2K JP2 JPE JPEG JPF JPG JPX MOS MRW NEF ORF PCT PDF PEF PIC PICT PNG PNTG PSD QIF QTI QTIF RAF RAW SGI SRF TGA TIF TIFF XBM

The plugin will only read an image if none of the other MeVisLab image plugins handle this image format, e.g. 3D-TIFF files will be accessed using the dedicated TIFF image reader of MeVisLab.

AppleScript support

AppleScript can be great for automating tasks, gluing applications together, or for extending the functionality of existing applications. The current edition of MeVisLab for OS X provides experimental support for AppleScript. To display the available scripting facilities open the scripting dictionary using the Script Editor's *Open Dictionary...* menu item (Fig. 7).

Figure 7 MeVisLab scripting dictionary displayed in Script Editor



For instance

```
tell application "Microsoft Excel"
  tell active sheet of active workbook
    --set paramsList to value of used range
    set paramsList to formula of used range
    repeat with currentParams in paramsList
      tell application "MeVisLab"
        set theResult to run macro "AppleScriptTest" with ¬
              parameters currentParams
      end tell
    end repeat
  end tell
end tell
```

will pass each row of contents of the current worksheet of a Microsoft Excel workbook as parameters to the function that has been assigned to the `consoleCommand` of the MeVisLab macro AppleScriptTest. Whereas

```
tell application "MeVisLab"
  repeat with incrementalValue from 1 to 10 by 2
    set theResult to run macro "AppleScriptTest" with ¬
           parameters {"Hello", incrementalValue}
  end repeat
end tell
```

will repeatedly run the macro AppleScriptTest by calling its `consoleCommand` with the given parameters. The line

```
set mlabVersion to version of application "MeVisLab"
```

will assign the current version of MeVisLab to the variable `mlabVersion`.

The following sections detail the OS X specific ways to launch MeVisLab and its associated utility applications.

Launching multiple MeVisLab instances

Launching an additional MeVisLab instance may be useful to separate the current work session from an experiment with new and potentially unstable MeVisLab modules or networks. To execute a new MeVisLab instance, choose *New Instance* from the context menu of the MeVisLab icon in the Dock (Fig. 8).

Figure 8 Launching a new MeVisLab instance via the context menu in the Dock

Quickstart MeVisLab

Quickstart minimizes the startup time of the IDE by skipping checks of the module database assuming that the module database did not change since the last startup of MeVisLab. In particular, MeVisLab will not look for new or changed Module Definition files (.def) in the module packages.

Besides quickstarting MeVisLab from the command line by adding the `-quick` option to the executable, one can quickstart MeVisLab interactively by pressing the Function (fn) key when launching MeVisLab from the Dock or while double-clicking on a MeVisLab Network document (.mlab).

Launching the ToolRunner

Besides launching the ToolRunner from the *File* menu of MeVisLab, two additional, more direct methods exist.

Adding the ToolRunner to the Dock

To add the ToolRunner icon to the Dock, launch the ToolRunner from the *File* menu of MeVisLab or via the *Tools* context menu of MeVisLab in the Dock (Fig. 9).

Figure 9 Launching the ToolRunner from the MeVisLab context menu in the Dock

Choose *Keep in Dock* from the Dock's context menu of the ToolRunner application to permanently glue the icon to the Dock (Fig. 10).

Figure 10 Adding the ToolRunner to the Dock

Starting the ToolRunner via Associated File

The ToolRunner supports several file types that are associated with specific tasks. Most of them prefix their file extension with the letters ml. Double-clicking on such a file, e.g. in the Finder or on the Desktop, will launch the ToolRunner and load the file.

Launching the MeVisLab Text Editor MATE

In the same way as the ToolRunner may be started independently of MeVisLab, MATE may be launched standalone via the Dock or by opening one of its supported file types via the Finder, e.g. a Module Definition .def file, as well.

To permanently add the MATE icon to the Dock, launch MATE from the *File* menu of MeVisLab or by selecting *Show Text Editor* from the *Tools* context menu of MeVisLab in the Dock. Then choose *Keep in Dock* from MATE's context menu (see Adding the ToolRunner to the Dock).

MeVisLab Release and Debug Mode

The Debug and Release versions of MeVisLab are provided within a single, self-contained OS X application bundle.

Characteristics of the MeVisLab variants

Each of the different MeVisLab variants has specific pros and cons that makes it more appropriate for certain tasks.

Release version

In Release mode, MeVisLab has been optimized for maximum performance and error tolerance. This is the mode best suited for executing mature and proven MeVisLab networks.

Maximum speed optimization and error tolerance come at the expense of verbose error reporting, however. For the development of MeVisLab networks and new modules, the Debug version is much better suited.

Debug version

The Debug version performs additional tests and is more verbose in certain conditions. Hence, this variant is well suited for the development of new MeVisLab networks. Also use the Debug version, if you are testing new modules or want to debug a module using Xcode (See *Module Development* on page 18).

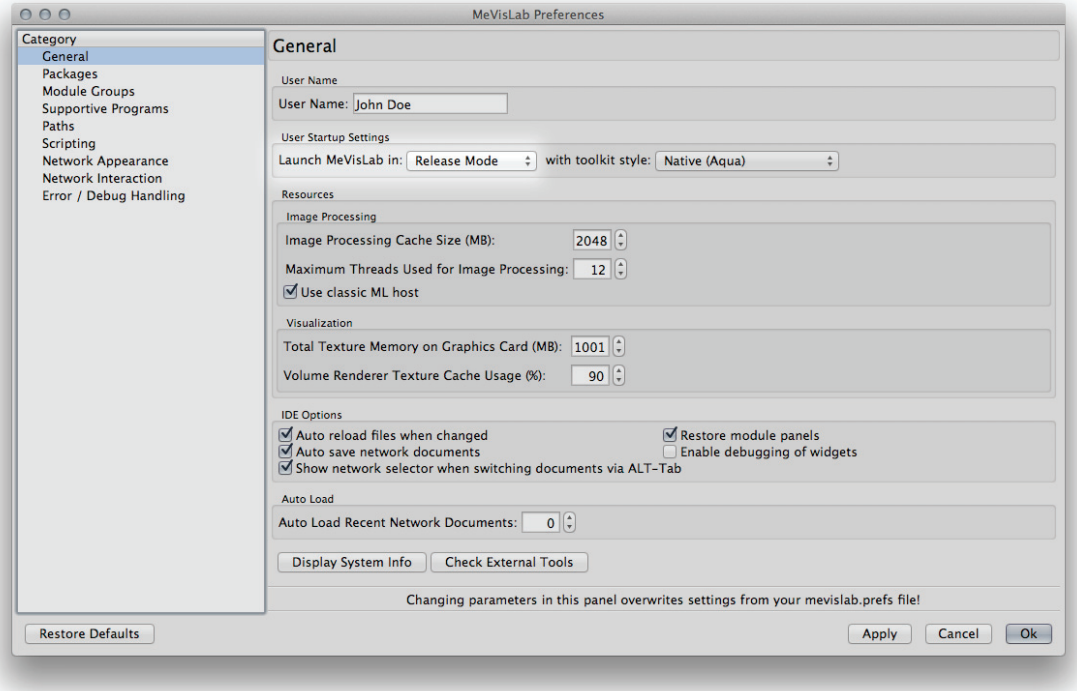
Due to the limited code optimization and additional runtime checks, the Debug version offers less performance in executing large MeVisLab networks.

How to start MeVisLab in Release or Debug mode

Currently, there are three options to control the selection of the started variant when you launch MeVisLab.

To permanently change the default startup variant, use the *MeVisLab › Preferences… › General › User Startup Settings* section in the Preferences panel of MeVisLab (Fig. 11).

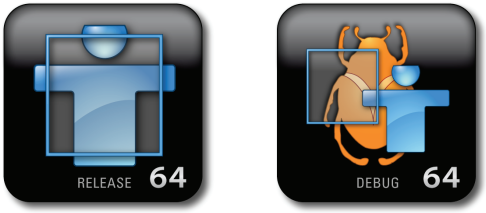
Figure 11 Choosing Debug or Release as startup variant of MeVisLab



To select an alternative variant for a single session, press the Option key on the keyboard while launching MeVisLab. You may use the Option key as well, if MeVisLab is launched indirectly via double-click on a MeVisLab network document icon (.mlab), e.g. on the Desk-

top or in a Finder window. The MeVisLab application icon in the Dock will change and display the icon of the started variant (Fig. 12).

Figure 12 Icons of the different MeVisLab variants



A specific MeVisLab variant may also be launched from the command line by passing an option to the MeVisLab starter executable. Valid choices are `-release` and `-debug`, e.g.

```
.../MeVisLab.app/Contents/MacOS/MeVisLab -release
```

to start the Release version, and

```
.../MeVisLab.app/Contents/MacOS/MeVisLab -debug
```

to launch MeVisLab in Debug mode.

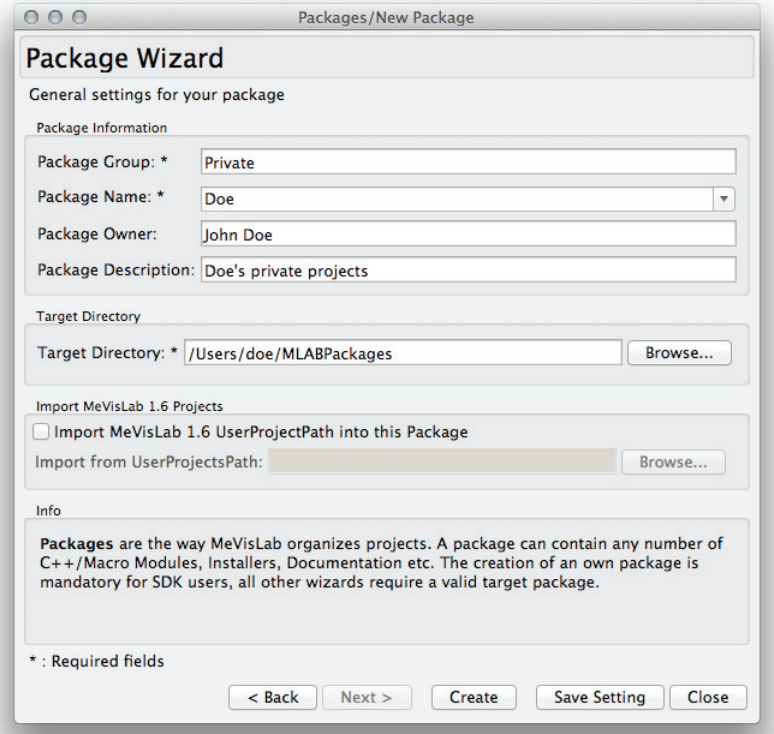
Module Development

Development of MeVisLab modules is supported using Xcode and UNIX Makefiles on the Mac. To facilitate the development of MeVisLab modules across different platforms, projects files are stored in a platform independent manner using Qt-Project's QMake project files (called .pro files).

Setup of a User Package for Your Modules

MeVisLab modules are organized in packages. A package may contain any number of modules including documentation and other resources. A package is self-contained to facilitate relocation and distribution. Before you start developing your own modules, you need to setup at least one user package. There is a package wizard that will assist you in setting-up a new package if you didn't already create one.

Figure 13 Setup of a new package using the package wizard



The module wizards can be accessed via the *File* menu by choosing *Run Project Wizard…*.

Using the Module Wizards to create a module

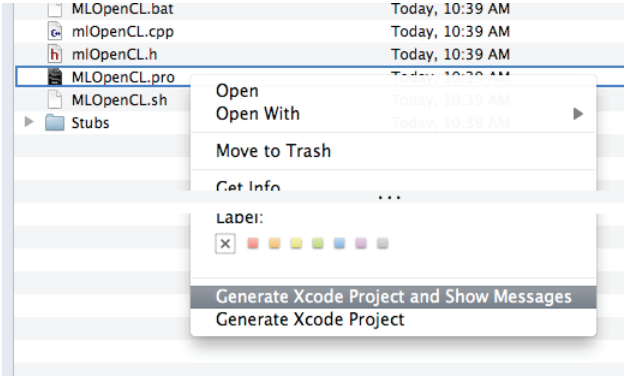
MeVisLab comes with powerful wizards that guide you through the process of module creation. See *Developing Modules* in the *Getting Started with MeVisLab* tutorial for further information.

Creating an Xcode project from a .pro file

The MeVisLab OS X edition includes a helper application (called `MeVisLabProjectGenerator.app`) that registers itself with the OS X Finder as the default handler for .pro files and also provides OS X Services related to .pro files. MeVisLab must be started at least once to perform this registration. Upon that, Xcode projects may be created from .pro files by double-clicking a projects .pro file in a Finder view of its folder. The contextual menu of the Finder's folder view will also offer two *Generate Xcode Project** menu items for a .pro file entry (Fig. 14). Depending on the number of services that offer items for the context menu, these menu items may end up in a separate *Services* sub-menu. You may enable or disable these contextual menu items in the *Keyboard Shortcuts* tab of the *Keyboard* system preferences pane.

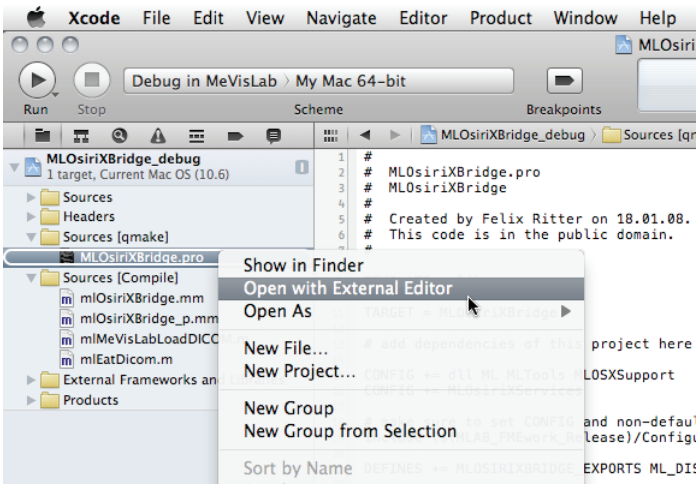
Sometimes you may want to inspect diagnostic messages from the project generation process, for instance to detect and to resolve dependency issues of your project. You may either hold down the Option-key while double clicking a .pro file or choose the *Generate Xcode Project and Show Messages* menu item in Finder or the *Services* menu to bring up a dialog that displays all messages.

Figure 14 Generating the Xcode project via the contextual menu of the .pro file in Finder



To update an existing project from within the running Xcode IDE, Control-click the .pro file in the *Project Navigator* and choose *Open with External Editor* from the contextual menu. (Fig. 15). This will launch the `MeVisLabProjectGenerator.app` from within Xcode.

Figure 15 Rebuilding the Xcode project from the .pro file within Xcode itself



The `MeVisLabProjectGenerator.app` may be also executed from the command line using the `open` command line call

```
open myproject.pro
```

or by launching the bundle executable directly. Type the following call

```
.../MeVisLab.app/Contents/Support/MeVisLabProject-
Generator.app/Contents/MacOS/MeVisLabProjectGen-
erator --help
```

in the Terminal to see all available options.

The helper application creates separate Xcode projects for release and debug builds. Use the `YourProject.xcodeproj` Xcode project to

build a code optimized version of your module in Release mode. If you wish to inspect your module in Debug mode, you must use the project `YourProject_debug.xcodeproj`. See *Debugging MeVisLab modules using Xcode* on page 22 for further information.

You may also create and open an Xcode project from within MeVisLab via the context menu of a MeVisLab module (*Edit Files › myproject.xcodeproj*). Since MeVisLab uses some heuristics to locate your project, this entry may not always be included in the menu.

Creating Makefiles from a .pro file

The `MeVisLabProjectGenerator.app` helper application may create UNIX Makefiles from .pro files as well. In GUI mode, however, when started via Finder or from Xcode itself, the `MeVisLabProjectGenerator.app` will only create Xcode project files. To generate Makefiles launch the `MeVisLabProjectGenerator.app` executable directly from the Terminal. Type the following call

```
.../MeVisLab.app/Contents/Support/MeVisLabProject-
Generator.app/Contents/MacOS/MeVisLabProjectGen-
erator --help
```

in the Terminal to see all available options.

Debugging MeVisLab modules using Xcode

To simplify the setup of your Xcode project for the debugger, the MeVisLab Project Generator creates a *Debug in MeVisLab* scheme during the Xcode project creation step. If you activate this scheme and click *Run*, Xcode will launch the MeVisLab Debug version and enable you to load and debug your MeVisLab module.

Obtaining Source Codes of MeVisLab Modules

Your installation of the MeVisLab SDK contains the sources of selected modules and several additional example modules. Those example modules are located within the `MeVisLab.app` bundle in the `/Contents/Packages/MeVisLab/Examples/Sources` subdirectories. They are a great source of information. However, not all of the included source code projects are ready to compile. Some of the projects require modifications to the project settings and/or additional files that may not be included.

MeVisLab Keyboard Shortcuts

The following sections list key or key modifier and mouse click combinations that are specific to the OS X edition of MeVisLab.

Startup keyboard shortcuts

Key or key combination	What it does
Option-Click on Icon	Launch MeVisLab using an alternative Variant (Debug or Release)
Shift-Click on Icon	Launch MeVisLab using the alternative Toolkit Style (Application or Native)
Function-Click on Icon	Quickstart MeVisLab

Document Revision History

Date	Notes
2014-03-21	General update
2013-01-15	Installation updated Creating an Xcode project from a .pro file updated
2012-05-18	Creating an Xcode project from a .pro file updated Debugging MeVisLab modules using Xcode 3 removed
2011-06-20	Creating an Xcode project from a .pro file updated Debugging MeVisLab modules using Xcode 4 added
2010-12-01	Launching multiple MeVisLab instances added Launching the ToolRunner updated Launching the MeVisLab Text Editor MATE added Quickstart MeVisLab added Debugging MeVisLab modules with Xcode updated MeVisLab Keyboard Shortcuts added
2009-10-16	Table of Contents added Changing the Toolkit Style of MeVisLab updated AppleScript support added Creating an Xcode project from a .pro file updated

Date	Notes
2009-04-22	Additional Features of the MeVisLab IDE renamed to Mac-Specific Features of the IDE Launching the ToolRunner added Changing the Toolkit Style of MeVisLab added MeVisLab Release and Debug Mode added OS X Image IO integration added Configuration of the User Projects Location renamed to Setup of a User Package for Your Modules Creating an Xcode project from a .pro file updated Creating Makefiles from a .pro file updated Debugging MeVisLab modules with Xcode updated Developing on Leopard for Tiger removed Integrating MeVisLab with OsiriX moved to separate document
2008-03-10	PowerPC sections removed Spotlight importer plugin added Quick Look integration added Setup of User Projects Location added Using the Module Wizards added Creating an Xcode project from a .pro file updated Debugging MeVisLab modules with Xcode added Developing on Leopard for Tiger added Obtaining Source Codes of MeVisLab Modules added Integrating MeVisLab with OsiriX added
2007-04-09	New document that introduces specific features and modifications of the OS X edition of MeVisLab