# Contents

1. **Overview** .......................................................................................................................... 3
   1.1 Audience............................................................................................................................ 3

2. **Development Environment Overview** ............................................................................. 4
   2.1 Build Tools .......................................................................................................................... 4
   2.2 External Libraries ............................................................................................................... 4
   2.3 Additional Tools ................................................................................................................. 5

3. **Tool Installation and Setup** .............................................................................................. 6
   3.1 Visual Studio 2013 ............................................................................................................. 6
   3.2 WiX ................................................................................................................................... 6
   3.3 CMake ............................................................................................................................... 6
   3.4 Optimet SDK ..................................................................................................................... 6
   3.5 WxWidgets 3.0.2 ............................................................................................................... 7
   3.6 Open Source Computer Vision (OpenCV) ....................................................................... 7
   3.7 Mobile Robot Programming Toolkit ................................................................................ 7
   3.8 WxFormBuilder (Optional) .............................................................................................. 8

4. **Development Machine Setup** ........................................................................................... 10
   4.1 Environment Variables ...................................................................................................... 10
   4.2 Developer Key .................................................................................................................. 10

5. **Optimet Plugin Project** ...................................................................................................... 11
   5.1 Visual Studio Project ......................................................................................................... 11
   5.2 FormBuilderProjects ....................................................................................................... 12
   5.3 Mach4Macros ................................................................................................................... 12
1 Overview

The Optimet Plugin for Mach 4 brings the measurement capabilities of Optimet measurement devices to the Mach 4 platform.

1.1 Audience

This developer guide is intended for use by programmers who wish to modify and recompile the Optimet Plugin source code.

The Optimet Plugin is a Windows library with a few dependencies. In order to modify and recompile the source code, a developer needs the proper build tools and dependencies installed on their development machine.

This developer guide will explain:

- where to download all of the build tools and dependencies
- how to install and configure the build tools and dependencies
- configuration steps that must be executed on the development machine
- how to build the Optimet Plugin library from the Visual Studio project
2 Development Environment Overview

This section lists the tools and libraries required for the development environment. Links to download tools and libraries are provided below.

Version numbers are provided for reference. These were the versions used to build the plugin at the time this document was published. It is recommended to use the same version for future builds unless there is a new feature or bug fix required from a newer release.

The Mach 4 SDK is required for building, but it is not listed below. The Mach 4 SDK is not freely available to download. You will need to contact Mach 4 support to register as a developer in order to receive access to the SDK. The SDK is currently distributed as a zip file and can be extracted anywhere on the development machine.

2.1 Build Tools

The following tools are required to compile the plugin and other libraries:

- Visual Studio 2013 [Update 4]
- WiX [v3.10.2]
  http://wixtoolset.org/releases/
- CMake [v3.4.1]
  https://cmake.org/download/

2.2 External Libraries

The Optimet Plugin depends on the following external libraries.

- Optimet SDK (32-bit) [v1.2.11]
  http://www.optimet.com/smart32-sdk.php
- WxWidgets [v3.0.2]
  http://sourceforge.net/projects/wxwindows/files/3.0.2/
- Mobile Robot Programming Toolkit (MRPT) [v1.3.2]
  http://www.mrpt.org/download-mrpt/
- Open Source Computer Vision (OpenCV) [v3.0.0]
  http://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.0.0/opencv-3.0.0.exe/download
2.3 Additional Tools

- WxFormBuilder [v3.5.1-rc1]
  http://sourceforge.net/projects/wxformbuilder/
3 Tool Installation and Setup

3.1 Visual Studio 2013
Download and install Visual Studio 2013. You must include the C++ build tools with the installation.

The Optimet Plugin was developed with Visual Studio 2013 Premium Edition with Update 4 installed.

You should be able to use any version of Visual Studio 2013 at the Professional Edition level or better.

3.2 WiX
Download and install WiX (a toolkit for building Windows Installers) from http://wixtoolset.org/releases/. The default installation options are fine.

Be sure to complete this step after installing Visual Studio 2013 so that WiX installs Visual Studio extensions.

WiX is not required to build the plugin itself. However, it is required to build the installer-related projects in the solution. These projects include the Optimet installer, the Optimet installer bundle (which bundles the Visual C++ 2013 redistributable with the Optimet installer), and a project containing custom actions for the Optimet installer.

3.3 CMake
Download the Win32 installer for CMake from https://cmake.org/download/ (e.g. cmake-3.4.1-win32-x86.exe).

Note that new versions of CMake are released regularly, and older versions are sometimes hard to find. You should be able to use the latest 3.x version of CMake if 3.4.1 is no longer easily available.

Install CMake using the self-extracting installer. The default installation options are fine.

3.4 Optimet SDK

Install the SDK using the self-extracting installer. It will unpack the SDK into C:\Program Files (x86)\Optimet\Ethernet Probe SDK\Ethernet Probe SDK.

The libraries that you will need to link with the Optimet Plugin are located at C:\Program Files (x86)\Optimet\Ethernet Probe SDK\Ethernet Probe SDK\Code Samples\Bin.
3.5 **WxWidgets 3.0.2**

Download the WxWidgets installer for windows (wxMSW-3.0.2-Setup.exe) from http://sourceforge.net/projects/wxwindows/files/3.0.2/.

Run the installer, and unpack the contents to C:\Libraries\WxWidgets-3.0.2.

Build WxWidgets using Visual Studio:

- Open a Visual Studio Command Prompt**
- Change directory to %WXWIN%\build\msw
- Build the debug version with the following command:
  nmake /f makefile.vc
- Build the release version with the following command:
  nmake /f makefile.vc BUILD=release

**Note that a Visual Studio Command Prompt is NOT the same as a regular command prompt. The Visual Studio Command Prompt puts Visual Studio tools into the environment for the command line session. You must use a Visual Studio Command Prompt to build. If you cannot find a shortcut for this in your start menu, you can create a shortcut or add it to the list of external tools in Visual Studio. Search the MSDN documentation for details.

For more information about building, see C:\Libraries\wxWidgets-3.0.2\docs\msw\install.txt.

The libraries that you will need to link with the Optimet Plugin are located at C:\Libraries\wxWidgets-3.0.2\lib\vc_lib.

3.6 **Open Source Computer Vision (OpenCV)**

NOTE: The Mobile Robot Programming Toolkit depends on OpenCV. You must install the OpenCV binaries before installing and building MRPT.


Install the OpenCV libraries using the self-extracting installer. This document assumes the installation location is set to C:\Libraries\opencv-3.0.0.

The libraries you will need to use for building are located at C:\Libraries\opencv-3.0.0\opencv\build|x86|vc12.

3.7 **Mobile Robot Programming Toolkit**

Download the MRPT source code from http://www.mrpt.org/download-mrpt/. You must download the source code (e.g. mrpt-1.3.2.zip).

Extract the zip file. This document assumes the zip is extracted to C:\Libraries\mrpt-1.3.2.
NOTE: You must have completed all of the previous steps before continuing. MRPT depends on Visual Studio, CMake, WxWidgets, and OpenCV.

Follow these steps to create the Visual Studio project files required to build:

- Start CMake
- If you are prompted to choose a generator, select Visual Studio 12 (2013).
- Set the source directory to `C:/Libraries/mrpt-1.3.2`
- Set the build directory to `C:/Libraries/mrpt-1.3.2/install`
- Click Configure
- When finished, make the following changes to the settings:
  - Turn off BUILD_SHARED_LIBS (this will cause static libraries to be built)
  - Turn off COMPILE_IN_PARALLEL (this conflicts with using precompiled headers)
  - Set the OpenCV_DIR (`C:/Libraries/opencv-3.0.0/opencv/build` if you followed this guide)
- Click Generate to create the Visual Studio project files with the correct settings.

NOTE: You MUST build MRPT as static libraries. Because MRPT is “sharing” the same WxWidgets build as the plugin, MRPT must be statically linked to the plugin to execute correctly. Failure to change this setting will cause the plugin to misbehave at runtime.

NOTE: You MUST turn off COMPILE_IN_PARALLEL. Failure to change this setting will cause Visual Studio to fail with multiple errors. You cannot perform a parallel build when using precompiled headers.

Follow these steps to build the MRPT libraries required by the plugin:

- Open the Visual Studio solution (`C:\Libraries\mrpt-1.3.2\install MRPT.sln`).
  - NOTE: You may need to be patient while the solution loads.
  - You DO NOT NEED TO BUILD THE ENTIRE SOLUTION FROM THE TOP LEVEL. It is very time consuming and you will end up with almost 70 unused libraries! Follow the steps below!
  - The projects to build are located in the modules section of the master solution.
- Build the base, gui, and opengl projects in Debug mode. (Note that gui should build opengl as a dependency.) These projects may take a while to compile. Please be patient.
- Repeat the build for base, gui, and opengl in Release mode.

The libraries that you need to link with the plugin will be located at `C:\Libraries\mrpt-1.3.2\install\lib`.

### 3.8 WxFormBuilder (Optional)

WxFormBuilder is a WYSIWYG editor for designing WxWidgets windows. It includes a code generator that will translate the designed window into several languages, including C++.

If you are going to modify the GUI of the Optimet Plugin, you should install WxFormBuilder so that you can work with the existing WxFormBuilder project files. (See Section 5.2 for more details.)
Download the Windows installer for WxFormBuilder from

Install the application with the self-extracting installer. The default installation options are fine.
4 Development Machine Setup

You are almost ready to build the Optimet plugin. Before continuing, there are a couple more things to set up on the development machine.

4.1 Environment Variables

The Optimet Plugin’s Visual Studio project file uses environment variables to indicate paths for various header files and libraries. This makes it easier to migrate to new versions of the Mach4 SDK and new versions of external libraries.

To set environment variables, go to Control Panel > System and Security > System and choose Advanced system settings on the left. In the System Properties dialog, choose the Advanced tab and then Environment Variables… You can add the following variables as User variables.

**NOTE:** The values for these variables are examples, assuming that you have followed this guide. Please adjust these values to match your Mach4 SDK version and tool versions.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>M4HOBBY</td>
<td>C:\Customers\Optimet\Mach4_and_SDK-2849\Mach4</td>
</tr>
<tr>
<td>MACHSDK</td>
<td>C:\Customers\Optimet\Mach4_and_SDK-2849\Mach4\SDK</td>
</tr>
<tr>
<td>MRPTSDK</td>
<td>C:\Libraries\mrpt-1.3.2</td>
</tr>
<tr>
<td>OPENCVLIB</td>
<td>C:\Libraries\opencv-3.0.0\opencv\build\x86</td>
</tr>
<tr>
<td>OPTIMETSDK</td>
<td>C:\Program Files (x86)\Optimet\Ethernet Probe SDK\Ethernet Probe SDK\Code Samples</td>
</tr>
<tr>
<td>WXWIN</td>
<td>C:\Libraries\wxWidgets-3.0.2</td>
</tr>
</tbody>
</table>

4.2 Developer Key

The Optimet Plugin must be properly signed in order for the Mach 4 application to load it.

The Optimet Plugin’s Visual Studio project is configured to use the DangerLabs key for signing. The file with the key is named DangerLabs.prv.

Copy DangerLabs.prv to C:\Customers\Optimet\Mach4_and_SDK-2849\Mach4\DeveloperKeys. (You will need to create the DeveloperKeys directory.)

The Visual Studio project uses a post-build step to automatically sign the plugin with this key and copy the plugin and signature file to C:\Customers\Optimet\Mach4_and_SDK-2849\Mach4\Plugins.

**NOTE:** The plugin could be signed with a different key. To do so, you will need to change the post-build steps in the Visual Studio project file. Also, you will need to update the source code. Find the call to mcPluginRegister() inside of mcPluginLoad() and change the developerId argument to match the developer ID associated with the new key.
5 Optimet Plugin Project

5.1 Visual Studio Project

The Visual Studio solution for the Optimet Plugin is located at %ProjectHome%\SourceCode\OptimetPlugin\OptimetPlugin.sln.

Assuming all of the project dependencies have been built or installed (see Section 3), the Optimet Plugin is ready to build. Simply right-click the OptimetPlugin project and select Build to build it.

When the compiling and linking phase are finished, the project contains post-build instructions which perform the following steps:

- Copy the plugin into the Mach4 SDK’s Plugins directory.
- Create a signature (.sig) file and copy it to the Plugins directory.

NOTE: Visual Studio cannot copy over the .m4pw or .sig file if Mach 4 is running. Be sure you shut down Mach 4 before building!

If your development environment is properly configured, you should also be able to start the Mach 4 application from the Visual Studio IDE, set breakpoints, and debug the plugin. Note that the project is configured to automatically start a Mach 4 profile called “Optimet.” You can change this in the project settings under Configuration Properties > Debugging. (For example, you can remove the arguments completely, and Mach 4 will prompt you for the profile to load at startup. Otherwise, just change the profile name after /p to the profile you are using.)

NOTE: The Optimet Plugin for Mach 4 has a few library dependencies (Smart32.dll, opencv_world300.dll, NewProbeDialog.dll, and NTGraph.ocx) that must be “installed” into the SDK directory for the plugin to load and run. Please see the user guide if you need to know where to copy these files. This is a one-time setup step.

The Visual Studio Project contains a Debug and Release build. There is a big difference in performance and code size between Debug and Release. Make sure you build for Release if you are creating a new plugin for distribution.
5.2 FormBuilderProjects
The GUI for the Optimet Plugin was built using WxWidgets. This allows the plugin to have an integrated panel within a screen set. It also gives the plugin a head start toward cross-platform compatibility. (Mach 4 is designed to run on Linux and Mac OS X, too.)

The main plugin panel and all of the dialogs/windows for the plugin were designed with WxFormBuilder. This is a WYSIWYG editor for creating WxWidgets interfaces. All of the form builder projects (.fbp files) are located under %ProjectHome%\FormBuilderProjects.

Within the Visual Studio project, you will see that each dialog/panel (such as DlgConfiguration.h|cpp) has a related Design (such as DlgConfigurationDesign.h|cpp). The files with the Design designation were created from WxFormBuilder’s auto-generated code.

The Design files define the appearance of the window and define an interface to interact with buttons, check boxes, and other controls. THESE FILES ARE COPIED DIRECTLY FROM WxFormBuilder AND SHOULD NEVER BE MODIFIED BY HAND.

The non-Design files implement the behavior of the dialog/panel by overriding the interface. These files are intended to be modified by a developer.

If you wish to update an existing dialog or panel:

- Double-click the associated .fbp file to open it using WxFormBuilder.
- On the Designer tab, use the editor to make the required changes.
- When finished, go to the C++ tab.
- Select all of the contents of the generated cpp file, copy it, and paste it into the appropriate Design.cpp file.
- Select all of the contents of the generated .h file, copy it, and paste it into the appropriate Design.h file.
- Modify the non-Design files to implement new behavior, if needed.
- Build, test, and iterate over the above steps until finished.
- Check in the .fbp along with the other project changes.

5.3 Mach4Macros
Macro files that are distributed with the Optimet Plugin are kept under source control at %ProjectHome%\Mach4Macros. These files are Lua scripts that provide basic functionality for executing plugin commands during G Code. (For example, m100.mcs triggers the “start measurement stream” command.)

Building the application does not depend on these files in any way. Simply check out, update, and check in these macros as needed.