qMaZda Installation Guide

qMaZda does not include an installer. The software package must be installed manually by downloading and extracting the executable files into a newly created directory on your computer.

Installing on Windows Systems

To install qMaZda on a Windows system, first create a new folder named `qmazda` on your computer's `C:` drive. Then, download the file `qmazda2501_win64.zip` and extract its contents into the `qmazda` folder. Next, download the manual file `qmazda.pdf` and save it in the same folder. Finally, run the program by double-clicking `MaZda.exe`. You can now start using qMaZda.

Note: The Windows version includes precompiled versions of selected Qt libraries. These libraries are provided under open-source licensing terms and may be used with certain restrictions (https://www.qt.io/licensing/open-source-lgpl-obligations). Additionally, qMaZda utilizes the following open-source libraries: Libtiff, Alglib, OpenCV, InsightToolkit, Libqhull, and LibSVM. The Windows version of qMaZda was cross-compiled on the Kubuntu Linux system using x86_64-w64-mingw32 tools.

Known Issues and Troubleshooting

Programs Blocked by Windows or Antivirus Software

Some versions of Windows or antivirus software may block programs from unknown sources. This issue may manifest as the message *Failed to execute generator* or as difficulties launching tools such as *Report Editor* or *Map Viewer* from the *View* menu.

To resolve this issue, run each of the following programs from the `qmazda` folder manually before launching `MaZda.exe`: `MzGenerator`, `MzGengui`, `MzReport`, and `MzMaps`. When prompted, confirm that each program can be safely executed. If the issue persists, check your antivirus software settings to ensure it is not blocking these programs.

Missing Shared Libraries

Certain antivirus programs may remove unknown shared libraries (`.dll` files) automatically. This may cause the `MzMaps` or `MzReport` tools to lose functionality, such as the ability to perform analyses.

If you suspect this issue, check the `qmazda` folder to ensure that the files beginning with `mzp` and ending with `.dll` are present. If these files are missing, copy them back to the folder and configure your antivirus software to exclude these files from scans, preventing them from being deleted or blocked.

File Name Compatibility Issues

qMaZda may not properly read or write files whose names include special characters, spaces, or non-Latin alphabet letters. This limitation is due to the Insight Toolkit (ITK) library used by qMaZda, which requires filenames encoded in UTF-8. Unfortunately, Windows does not fully support filenames in UTF-8 (see Unicode in Microsoft Windows).

To resolve this issue, use file and folder names that contain only basic Latin letters (A-Z, a-z), numbers, and underscores. Avoid using spaces, special characters, or characters from non-Latin alphabets.

Alternatively, you can enable UTF-8 support in Windows. To do this, open *Settings and navigate* to *Time & Language*, then select *Region*. Under *Administrative language settings*, choose *Change system locale* and check the box labeled *Use Unicode UTF-8 for worldwide language support*. Note that enabling UTF-8 support may affect the behavior of other programs on your system.

Installing on Linux Systems

To install qMaZda on a Linux system, create a folder in your home directory with any name, preferably `qmazda`. Download the file `qmazda2501lite_amd64.tar.gz` and extract its contents into the folder you created. Then, download the manual file `qmazda.pdf` and save it in the same folder.

Next, install the libraries required for qMaZda to run. These include Qt5, Libtiff, Alglib, OpenCV, InsightToolkit, Libqhull, and LibSVM. Open a terminal and execute the following commands to install these dependencies:

```
sudo apt install qtbase5-dev
sudo apt install libtiff-dev
sudo apt install libalglib-dev
sudo apt install libopencv-dev
sudo apt install libinsighttoolkit5-dev
sudo apt install libqhull-dev
sudo apt install libsvm-dev
```

Once the libraries are installed, run the program `MaZda` and enjoy using it.

Known Issues and Troubleshooting

Executable Permissions

If the program `MaZda` does not start, check whether the file is marked as executable. If it is not, modify its permissions by using your file manager to enable the *is executable* option, or use the `chmod` command in the terminal:

```
chmod +x MaZda
```

Also, ensure that the following files are marked as executable: `MzGenerator`, `MzTrainer`, `MzMaps`, `MzPredict`, `MzReport`, and `MzGengui`.

Blocked Programs

The Linux system may block programs originating from unknown sources. This issue may manifest as the error *Failed to execute generator* or difficulty accessing tools like *Report Editor* or *Map Viewer* from the *View* menu.

To resolve this issue, manually run the following programs from the `qmazda` folder before launching `MaZda`: `MzGenerator`, `MzGengui`, `MzReport`, and `MzMaps`. Confirm each program's execution when prompted to ensure they can run safely.

Incompatible Library Versions

Some versions of Alglib, OpenCV, InsightToolkit, Libqhull, or LibSVM libraries provided by your system's repository may not be compatible with qMaZda. To check for compatibility issues, start `MaZda` from the terminal and observe any error messages that appear. You can also use the `ldd` command to determine which libraries are required:

```
ldd MaZda
```

If errors indicate problems with these libraries, consider using the alternative software package `qmazda2501_amd64.tar.gz` instead of `qmazda2501lite_amd64.tar.gz`.

Building from Sources

If necessary, the qMaZda software package can be built from its source code. To do this, download the `qmazda2501_src.zip` file or clone the repository from https://gitlab.com/qmazda/qmazda. Extract the files to a folder on your computer. In the top directory of the extracted files, you will find a configuration file named `CMakeLists.txt`. This file is used to configure the project using the CMake tool (see: CMake Tutorial).

The qMaZda package requires several libraries, dependencies, to build successfully: Qt (version 5 or 6), Libtiff, Alglib, OpenCV, InsightToolkit, and Libqhull. Download the source code for these libraries and compile them before building qMaZda. Alternatively, you can install the `dev` versions of these libraries from your system's repository using `apt`, as described earlier, or download their precompiled versions from the official websites of the respective libraries.

Known Issues

Libqhull Compatibility

The version of the Libqhull library from the system repository may fail to link with the `mzpVschPlugin` library. In such cases, the linker will issue a warning indicating that Libqhull needs to be recompiled with the `-fPIC` compiler flag. To resolve this issue:

1. Download the source code for Libqhull.
2. Recompile it with the `-fPIC` flag enabled.
3. Use this custom-built version of Libqhull in your qMaZda project.

Alternatively, if the convex hull classifier functionality provided by the `mzpVschPlugin` library is not required, you can choose to exclude it from the build. To do this:

1. Open the `CMakeLists.txt` file in the main project directory.
2. Comment out or remove the line `add_subdirectory(VschPlugin)`.
3. Reconfigure the project using CMake and proceed with the build process.

Instalacja programów qMaZda

Pakiet programów qMaZda nie posiada instalatora. Instalacja pakietu wymaga ręcznego pobrania i rozpakowania plików do nowo utworzonego katalogu.

Instalacja w systemach Windows

Na dysku `C:` utwórz nowy katalog o nazwie `qmazda`. Pobierz plik `qmazda2501_win64.zip` i rozpakuj jego zawartość do utworzonego katalogu. Pobierz plik `qmazda.pdf` i zapisz go w tym samym katalogu. Uruchom program `MaZda.exe` i ciesz się jego funkcjonalnościami.

Uwaga: Wersja dla systemów Windows obejmuje skompilowane wersje wybranych bibliotek Qt. Biblioteki udostępnione są na zasadach licencji otwartego oprogramowania i mogą być wykorzystywane w ograniczonym zakresie (https://www.qt.io/licensing/open-source-lgpl-obligations). Ponadto, qMaZda korzysta z następujących bibliotek dostępnych na zasadzie otwartego oprogramowania: Libtiff, Alglib, OpenCV, InsightToolkit, Libqhull oraz LibSVM. Wersja qMaZda dla systemu Windows została przygotowana w systemie Kubuntu Linux za pomocą narzędzi x86_64-w64-mingw32.

Znane problemy

Blokowanie przez system lub program antywirusowy

Niektóre systemy Windows lub oprogramowanie antywirusowe mogą blokować uruchamianie programów z nieznanych źródeł. Objawia się to komunikatem *Failed to execute generator* lub trudnością w uruchamianiu funkcji z menu *View* > *Report editor...* lub *View* > *Map viewer...*.

Aby rozwiązać ten problem, przed uruchomieniem programu `MaZda` uruchom kolejno następujące programy: `MzGenerator.exe`, `MzGengui.exe`, `MzReport.exe` i `MzMaps.exe`. Podczas uruchamiania każdego z tych programów potwierdź, że program może być bezpiecznie uruchamiany.

Usuwanie bibliotek współdzielonych DLL przez antywirus

Niektóre programy antywirusowe mogą usuwać pliki DLL nieznanych bibliotek, co może prowadzić do problemów z działaniem programów takich jak `MzMaps` i `MzReport`. Objawia się to brakiem funkcji analizy.

Aby rozwiązać ten problem:

1. Sprawdź, czy w katalogu `qmazda` znajdują się pliki DLL, których nazwy zaczynają się od `mzp` i kończą się na `.dll`.
2. Jeśli ich brakuje, skopiuj je ponownie z oryginalnego archiwum.
3. Skonfiguruj program antywirusowy tak, aby nie usuwał ani nie blokował tych plików.

Problemy z nazwami plików

Program `MaZda` może nie obsługiwać prawidłowo plików o nazwach zawierających znaki specjalne lub litery narodowe. Wynika to z faktu, że biblioteka Insight Toolkit używana przez program korzysta z kodowania UTF-8, które nie jest w pełni obsługiwane w systemach Windows (więcej na ten temat: Unicode w Windows).

Aby uniknąć problemów:

1. Używaj plików o nazwach, które nie zawierają spacji, znaków specjalnych ani liter narodowych.
2. Dotyczy to również nazw katalogów, w których znajdują się pliki.

Alternatywnie możesz spróbować włączyć obsługę nazw plików w formacie UTF-8:

1. W ustawieniach systemu Windows przejdź do *Region & language*.
2. Wybierz opcję *Administrative* i kliknij *Change system locale...*
3. Zaznacz opcję *Use Unicode UTF-8 for worldwide language support*.

Instalacja w systemach Linux

Aby zainstalować qMaZda na systemie Linux, utwórz folder w katalogu domowym o dowolnej nazwie, najlepiej `qmazda`. Pobierz plik `qmazda2501lite_amd64.tar.gz` i rozpakuj jego zawartość do utworzonego folderu. Następnie pobierz plik `qmazda.pdf` i zapisz go w tym samym folderze.

Zainstaluj wymagane biblioteki, od których zależy działanie programu qMaZda. Należą do nich: Qt5, Libtiff, Alglib, OpenCV, InsightToolkit, Libqhull oraz LibSVM. Aby je zainstalować, otwórz terminal i wykonaj poniższe polecenia:

```
sudo apt install qtbase5-dev
sudo apt install libtiff-dev
sudo apt install libalglib-dev
sudo apt install libopencv-dev
sudo apt install libinsighttoolkit5-dev
sudo apt install libqhull-dev
sudo apt install libsvm-dev
```

Po zainstalowaniu bibliotek uruchom program `MaZda` i ciesz się jego funkcjonalnością.

Znane problemy i sposoby ich rozwiązania

Brak uprawnień do uruchomienia

Jeśli program `MaZda` nie uruchamia się, sprawdź, czy plik został oznaczony jako wykonywalny. Jeśli nie, zmień jego ustawienia za pomocą menedżera plików, zaznaczając opcję *Wykonywalny* lub użyj polecenia `chmod` w terminalu:

```
chmod +x MaZda
```

Upewnij się również, że pliki `MzGenerator`, `MzTrainer`, `MzMaps`, `MzPredict`, `MzReport` i `MzGengui` mają ustawione uprawnienia do wykonania.

Blokowanie programów przez system

System Linux może blokować uruchamianie programów pochodzących z nieznanych źródeł. Problem objawia się komunikatem *Failed to execute generator* lub trudnościami z dostępem do narzędzi, takich jak *Report Editor* lub *Map Viewer* z menu *View*.

Aby rozwiązać ten problem, uruchom ręcznie następujące programy z katalogu `qmazda` przed uruchomieniem `MaZda`: `MzGenerator`, `MzGengui`, `MzReport` i `MzMaps`. Przy każdorazowym uruchamianiu potwierdź, że programy te mogą działać bezpiecznie.

Niekompatybilne wersje bibliotek

Niektóre wersje bibliotek Alglib, OpenCV, InsightToolkit, Libqhull lub LibSVM dostępne w repozytorium systemowym mogą być niezgodne z wymaganiami programu qMaZda. Aby zweryfikować problemy z bibliotekami, uruchom `MaZda` z terminala i sprawdź pojawiające się komunikaty błędów. Możesz także użyć polecenia `ldd`, aby sprawdzić, jakie biblioteki są wymagane przez program:

```
ldd MaZda
```

Jeśli błędy wskazują na problemy z tymi bibliotekami, rozważ użycie alternatywnego pakietu programów `qmazda2501_amd64.tar.gz` zamiast `qmazda2501lite_amd64.tar.gz`.

Budowanie ze źródeł

Jeśli zajdzie potrzeba, programy z pakietu qMaZda można zbudować z kodów źródłowych. Pobierz i rozpakuj plik `qmazda2501_src.zip` lub sklonuj kody źródłowe z repozytorium dostępnego pod adresem https://gitlab.com/qmazda/qmazda. W głównym katalogu projektu `qmazda` znajdziesz plik konfiguracyjny `CMakeLists.txt`, który służy do konfiguracji projektu za pomocą programu CMake (Instrukcja użycia CMake).

Zwróć uwagę, że do kompilacji pakietu qMaZda wymagane są następujące biblioteki: Qt (wersja 5 lub 6), Libtiff, Alglib, OpenCV, InsightToolkit orazLibqhull. Pobierz kody źródłowe wymaganych bibliotek i skompiluj je przed rozpoczęciem budowy pakietu qMaZda. Alternatywnie możesz zainstalować wersje deweloperskie bibliotek z repozytoriów systemowych za pomocą komendy `apt`, jak opisano to w sekcji dotyczącej instalacji w systemie Linux, lub pobrać gotowe wersje bibliotek ze stron internetowych ich twórców. Po przygotowaniu bibliotek skonfiguruj projekt za pomocą narzędzia CMake, a następnie przejdź do kompilacji pakietu qmazda.

Znane problemy

Problem z biblioteką Libqhull

Wersja biblioteki Libqhull z repozytoriów systemowych może nie być zgodna z wymogami pakietu qMaZda. Podczas konsolidacji (linkowania) może pojawić się komunikat błędu informujący, że należy ponownie skompilować bibliotekę z opcją *-fPIC*.

Aby rozwiązać ten problem:

1. Pobierz kod źródłowy biblioteki Libqhull.
2. Skompiluj bibliotekę z włączoną opcją *-fPIC*.
3. Użyj tej wersji biblioteki w projekcie.

Alternatywnie, jeżeli klasyfikator z powierzchnią wypukłą nie jest Ci potrzebny, możesz pominąć kompilację wtyczki `mzpVschPlugin`. W tym celu otwórz główny plik CmakeLists.txt I zakomentuj lub usuń linię `add_subdirectory(VschPlugin)`. Ponownie skonfiguruj projekt za pomocą programu CMake. Po wykonaniu tych czynności możesz kontynuować budowę projektu i kompilację pakietu qMaZda.