**PROGRAMMERS INSTALLATION MANUAL**

**Software that are required:**

|  |  |  |
| --- | --- | --- |
| S No. | Software | Link |
| 1 | Python 2.7.3 or higher | <https://www.python.org/downloads/windows/> |
| 2 | Microsoft Visual C++ Compiler 9.0 for Python 2.7 | <https://www.microsoft.com/en-us/download/details.aspx?id=44266> |
| 3 | Qt Creator v4.0.1(based on Qt 5.6.1) | [qt-opensource-windows-x86-mingw492-5.6.1.exe](https://download.qt.io/official_releases/qt/5.6/5.6.1/qt-opensource-windows-x86-mingw492-5.6.1.exe) |
| 4 | OpenCV 2.4.3 or higher | <https://opencv.org/releases/> |
| 5 | CMake 3.10.win64-x64.msi |  |
| 6 | PostgreSQL 9.3 | <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads#windows> |
| 7 | GDAL | <http://gisinternals.com/> |
| 8 | Python wheel files | <https://www.lfd.uci.edu/~gohlke/pythonlibs/> |

**Installing Python and bindings:**

Install Python 2.7 (minimum version Python 2.7.3 and 32 bit) **into the default path (otherwise, you will have to change a lot of paths in the project) and include that path** (i.e., *“C:\Python27\”*) to PATH environment variable.

Download and install Visual C++ Compiler for Python 2.7 from MS VC++ Compiler.

Install the following Python bindings using pip: GDAL, matplotlib, numpy, pygame, PyQt4, scikit-image, scikit-learn, scipy, secure-smtplib, affine, imutils, pandas, psycopg2,cx\_freeze. To install the bindings, there are two ways:

* First and recommended way is to type *“C:\Python27\Scripts\pip install <binding-name>”* in the command prompt. This will install the most compatible version for your Python.
* Otherwise, if the first one does not work, go to this page Python Extension Packages for Windows - Christoph Gohlke and download the corresponding wheel file for the binding you want to install. Make sure you download the proper version matching the Python version with your Python installed. Then open command prompt in the folder where you have downloaded the wheel file and type *“C:\Python27\Scripts\pip install <binding-name-with-extension>”*. This will also install them.

## Installing Qt Creator:

* Install Qt Creator from the following link with the **default settings**.
* Link to install: [qt-opensource-windows-x86-mingw492-5.6.1.exe](https://download.qt.io/official_releases/qt/5.6/5.6.1/qt-opensource-windows-x86-mingw492-5.6.1.exe)
* Add *“C:\Qt\Qt5.6.1\Tools\mingw492\_32\bin\”* and *“C:\Qt\Qt5.6.1\5.6\mingw49\_32\bin\”* to PATH environment variable.
* Create a new variable named **QT5\_ROOT\_PATH** and set the value to *“C:\Qt\Qt5.6.1\5.6\mingw49\_32\bin\”*.

## Installing PostgreSQL and PostGIS:

* Install “**PostgreSQL 9.3 64 bit version**” into the **default path (please try to install it in the default path)**.
* While installing PostgreSQL, try to keep both **username and password as “postgres” and the port 5432** for convenience.
* After installing PostgreSQL, install only PostGIS2.2 bundle for your system from Stack Builder.
* Add folders *“C:\Program Files\PostgreSQL\9.3\bin”* and *“C:\Program Files\PostgreSQL\9.3\lib”* to PATH environment variable.

## Database Manual:

To run Dharohar application with the database, make sure the database server accepts remote connections. To access the database server remotely, the network address of the server must be mentioned in the **pg\_hba.conf** file.

One more thing that you need to be sure of is that appropriate PostGIS Bundle for PostgreSQL database should be installed on the machine to make Dharohar work properly.

### Create new database:

After creating a new database, run the SQL script “setupDatabase.sql” on the database. You can run it on the postgress shell /pg admin or command line. To run the script from command line, follow the syntax below.

*psql –U “username” -d “database\_name> -h “host\_address” -p “port” < setupDatabase.sql*

setupDatabase.sql file is attached with this document.

To run it from the shell or PG Admin, copy the contents of the file an paste it in the shell.

Provide the password of the user on prompt.

### Creating Schema:

Open the shell of the postgress and log into it with proper credentials.

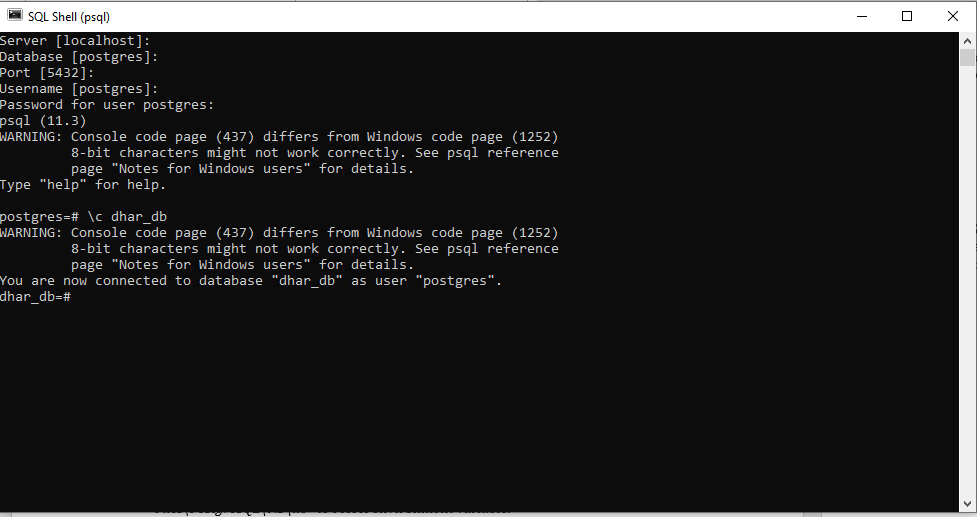
Server:

Database: dhar\_db

Port:5432

Username: postgresPassword: postgres

To connect to the database, use the command \c dhar\_db; and press enter key.



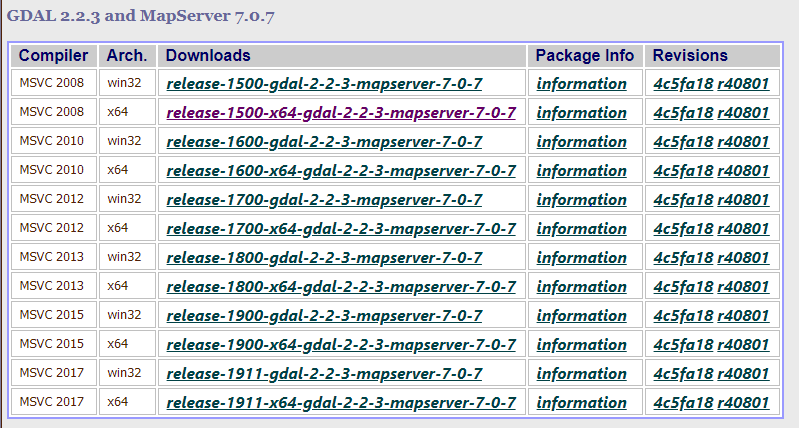
Copy the contents of the file schema.txt and then paste in the shell and press enter.

## **GDAL INSTALLATION:**

Install the requied gdal according to your processesor requirements.

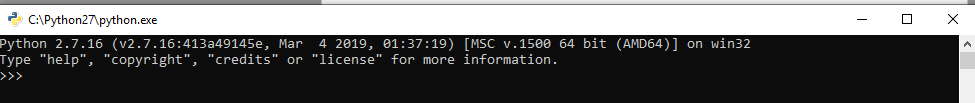
Gdal can be download from the website <http://gisinternals.com/>

Go to **Archive Versions**:



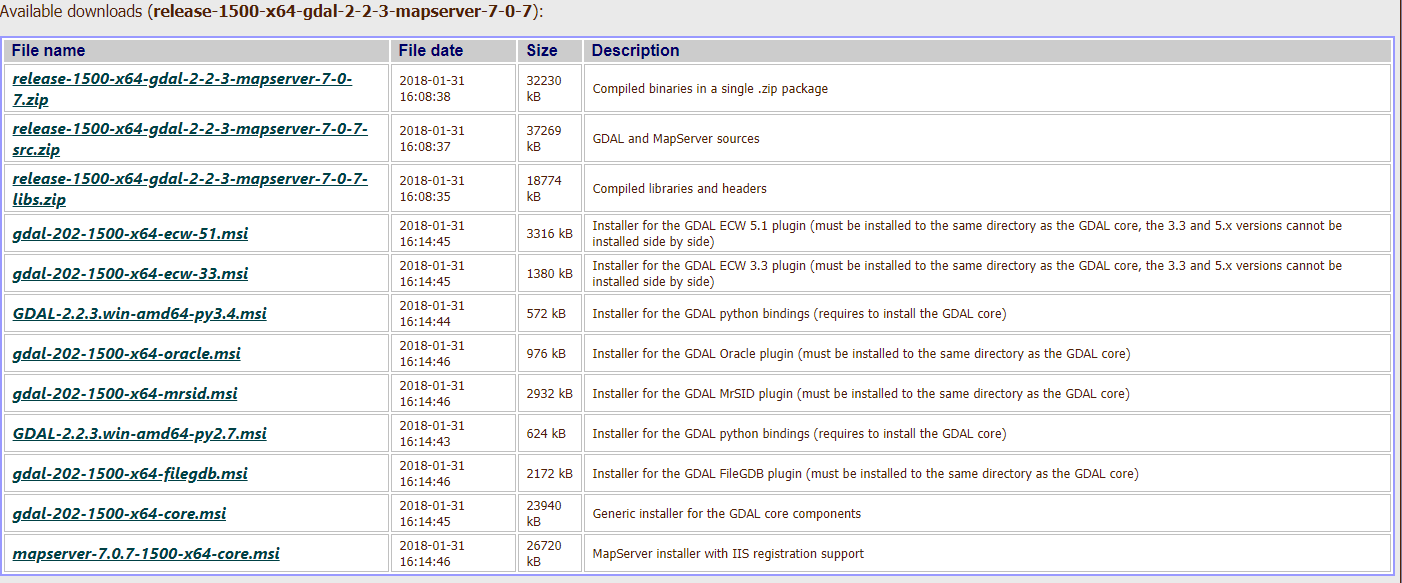
**Note:** MSC v.1500 may differ if you are using a different Python installation, if it does then please make a note of that number. Note, if you installed the 64-bit version of Python, for the rest of the tutorial **please remove the (x86)** from the paths.

To check the MSC run python (command line) or run IDLE in python.



We are using the MSC v.1500 on a 32-bit system, the picture below illustrates how to match the version with your own python version. The Arch. is where you should look for either 64-bit or 32-bit systems, and the Downloads shows the release-1500 number which should match the number from IDLE.

Clicking the link will take you to the list of binaries (installers) to download



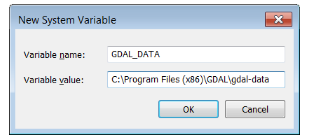
Locate the “core” installer, which has most of the components for GDAL.

After downloading your version, install GDAL with standard settings.

|  |  |
| --- | --- |
| Add 1 | ;C:\Program Files (x86)\GDAL |

to PATH environment variable.

Then create a new variable named **GDAL\_DATA** and set the value to C:\Program Files\GDAL\gdal-data.



Add one more new variable by clicking “New…” Add the following in the dialogue box:  
Variable name: GDAL\_DRIVER\_PATH  
Variable value: C:\Program Files (x86)\GDAL\gdalplugins  
Click “OK”

**NOTE: Proper installation of postgreSql, postgis and gdal is required for saving purpose. Without these softwares saving will not happen.**

**Installing OpenCV and configuring with CMake:**

There are two ways to do this, the first one is recommended. Please try to proceed with the first procedure.

**Recommended procedure:**

Run OpenCV and extract it to **C drive**.

**To configure OpenCV for Python,** copy file *cv2.pyd* from *“C:\opencv\build\python\2.7\<python architecture installed>\”* to *“C:\Python27\Lib\site-packages\”* and to **C drive**.

Copy contents of *opencv-build* folder to OpenCV’s build directory (i.e., *“C:\opencv\build\”*) folder. These are already build files hence, no need to build again.

Add *bin* directory under OpenCV’s build directory (i.e., *“C:\opencv\build\bin\”*) to PATH environment variable.

**OR,**

**Alternate procedure:**

Run OpenCV and extract it to **C drive**.

**For Language Translation:**

First we need to create a .ts file for all the .ui file and with that .ts file we can create .qm script file which contain the translation script using QtLinguistics.

**For qt:**

Open cmd where .pro file is present and then run the following command:

*lupdate –pro <pro file name>.pro -ts <language name>.ts*

using QtLinguistics open .ts file and then translation for each word in the respective language.

After translating all the word release the file it will create .qm file script.

**For python:**

Create a pro file which contain all python .ui files

Open cmd where .pro file is present and then run the following command:

*pylupdate4 –pro <pro file name>.pro -ts <language name>.ts*

**To build the program, copy the opencv build folder to the same folder where the source files folder is present**

*pylupdate4* is used as we are using pyqt4.

**The Dharohar software can be executed from the main.py file inside the untitled folder then the process will continue.**

**All the python files are inside the source folder.**

**Without Gdal saving to database will not happen. Proper installation of postgreSQL, Postgis and Gdal is required for the database process.**

**To configure OpenCV for Python,** copy file *cv2.pyd* from *“C:\opencv\build\python\2.7\<python architecture installed>\”* to *“C:\Python27\Lib\site-packages\”* and to **C drive**.

Install CMake with **default settings** and add it to the PATH environment variable.

Run CMake and select OpenCV’s *sources* directory (i.e., *“C:\opencv\sources”*) in the ***source*** section and select OpenCV’s *build* directory (i.e., *“C:\opencv\build”*) in the ***build*** section.

Configure and then Generate.

Add *bin* directory under OpenCV’s *build* directory (i.e., *“C:\opencv\build\bin\”*) to PATH environment variable.