Python

From Termux Wiki
Jump to: navigation, search

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

In Termux Python v3.x can be installed by executing

```
pkg install python
```

Legacy, deprecated version 2.7.x can be installed by

```
pkg install python2
```

Warning: upgrading major/minor version of Python package, for example from Python 3.8 to 3.9, will make all your currently installed modules unusable. You will need to reinstall them. However upgrading patch versions, for example from 3.8.1 to 3.8.2, is safe.

Note that Termux does not provide a way for downgrading. If you are not sure whether new Python version is appropriate for you, make a backup of $PREFIX (/wiki/Backing_up_Termux).

Package management

After installing Python, `pip` (`pip2` if using python2) package manager will be available. Here is a quick tutorial about its usage.

Installing a new Python module:

```
pip install {module name}
```

Uninstalling Python module:

```
pip uninstall {module name}
```

Listing installed modules:

```
pip list
```

When installing Python modules, it is highly recommended to have a package `build-essential` to be installed - some modules compile native extensions during their installation.

Python module installation tips and tricks

It is assumed that you have `build-essential` or at least `clang`, `make` and `pkg-config` installed.

It also assumed that `termux-exec` is not broken and works on your device. Environment variable `LD_PRELOAD` is not tampered or unset. Otherwise you will need to patch modules' source code to fix all shebangs!

Package	Description	Dependencies	
electrum	Lightweight Bitcoin wallet. https://electrum.org/ (https://electrum.org/)		`pkg install unstable-repo` `pkg install electrum`
gmpy2	C-coded Python modules for fast multiple-precision arithmetic. https://github.com/aleaxit/gmpy (https://github.com/aleaxit/gmpy)	libgmp libmpc libmpfr	
lxml	Bindings to libxml2 and libxslt. https://lxml.de/ (https://lxml.de/)	libxml2 libxslt	
Numpy	The fundamental package for scientific computing with Python https://numpy.org/ (https://numpy.org/)		`pip install numpy`

matplotlib	A plotting library for Python. https://matplotlib.org/ (https://matplotlib.org/)	freetype libpng	On some devices a patch/config file is need
			`pip install matplotlib` does not work,
			
			`git clone https://github.com/matplo`
			`cd matplotlib`
			`sed 's@#enable_lto = True@enable_lt`
			`pip install .`

pandas	Flexible and powerful data analysis / manipulation library for Python. https://pandas.pydata.org/ (https://pandas.pydata.org/)		`export CFLAGS="-Wno-deprecated-decl` `pip install pandas`
pynacl	Bindings to the Networking and Cryptography library. https://pypi.python.org/pypi/PyNaCl (https://pypi.python.org/pypi/PyNaCl)	libsodium	
pillow	Python Imaging Library. https://pillow.readthedocs.io/en/stable/ (https://pillow.readthedocs.io/en/stable/)	libjpeg-turbo libpng	
pyzmq	Bindings to libzmq. https://pyzmq.readthedocs.io/en/latest/ (https://pyzmq.readthedocs.io/en/latest/)	libzmq	

Advanced installation instructions

Some Python modules may not be easy to install. Here are collected information on how to get them available in your Termux.

Numpy and Scipy

Building complex software like numpy and scipy is tedious. Therefore, Termux user its-pointless (aka live_the_dream) (https://github.com/its-pointless) has packaged this software and maintain a Termux APT repository with these and many other useful packages.

Before Numpy/Scipy installation, you need to subscribe to APT repository:

```
curl -LO https://its-pointless.github.io/setup-pointless-repo.sh
bash setup-pointless-repo.sh
```

Then you can install Numpy or Scipy like a regular Termux package:

```
pkg install numpy
pkg install scipy
```

OpenCV

Instructions taken from #512 (https://github.com/termux/termux-packages/issues/512) and #1992 (https://github.com/termux/termux-packages/issues/1992). OpenCV is not a Python package but it includes the Python bindings (known as opencv-python in pip).

OpenCV needs to be built from source using CMake, install it and other dependencies with:

```
pkg install build-essential cmake libjpeg-turbo libpng python
```

Numpy is also required, see instructions for installing it above.

There might be other required dependecies as well, see the OpenCV docs (https://docs.opencv.org/trunk/d7/d9f/tutorial_linux_install.html) for the list.

The rest of the instructions can be copy-pasted straight away, but if you are not sure if you have all dependencies then it might be best to do it in two steps: first all commands up until the `LDFLAGS=" -llog" cmake` command and then in a second step make and make install.

To get the sources, git clone (from a suitable folder):

```
git clone https://github.com/opencv/opencv
cd opencv
```

You should now be in the opencv folder. Let's create a build folder where we will build the package:

```
mkdir build
cd build
```

To configure the package for python3 but not python2 (change the on/off flags to use python2 instead of python3) we run:

```
LDFLAGS=" -llog -lpython3" cmake -DCMAKE_BUILD_TYPE=RELEASE -DCMAKE_INSTALL_PREFIX=$PREFIX -DBUILD_opencv_
python3=on -DBUILD_opencv_python2=off -DWITH_QT=OFF -DWITH_GTK=OFF ..
```

Last command will throw errors if there are missing dependencies. After this we can compile the package with

```
make
```

and then install the files with

```
make install
```

Tkinter

Tkinter is splitted of from the `python` package and can be installed by

```
pkg install python-tkinter
```

We do not provide Tkinter for Python v2.7.x.

Since Tkinter is a graphical library, it will work only if X Windows System environment is installed and running. How to do this, see page Graphical Environment (/wiki/Graphical_Environment).

Installing Python modules from source

Some modules may not be installable without patching. They should be installed from source code. Here is a quick how-to about installing Python modules from source code.

1. Obtain the source code. You can clone a git repository of your package:

```
git clone https://your-package-repo-url
cd ./your-package-repo
```

or download source bundle with `pip` :

```
pip download {module name}
unzip {module name}.zip
cd {module name}
```

2. Optionally, apply the desired changes to source code. There no universal guides on that, perform this step on your own.

3. Optionally, fix the all shebangs. This is not needed if `termux-exec` is installed and works correctly.

```
find . -type f -not -path '*/\.*' -exec termux-fix-shebang "{}" \;
```

4. Finally install the package:

```
python setup.py install
```