

RUTM50 Python3

[Main Page](#) > [RUTM Routers](#) > [RUTM50](#) > [RUTM50 Manual](#) > [RUTM50 WebUI](#) > [RUTM50 Services section](#) > **RUTM50 Python3**

The information in this page is updated in accordance with firmware version [RUTM_R_00.07.07.1](#).

□

Contents

- [1 Summary](#)
- [2 Python3 Usage](#)
- [3 Python3 Modules](#)

Summary

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

Teltonika **Python3** package uses Python version **3.9.7**. The list of all modules included in the package can be found bellow.

This manual page provides an overview of Python3 functionality in RUTM50 devices.

Note: Python is additional software that can be installed from the **System** → [Package Manager](#) page.

Disclaimer: before installing Python3 package make sure that the target device has a sufficient amount of free storage space!!!

Python3 Usage

After installing the package a new command will become available in [CLI \(Command-line interface\)](#) which enables the device to invoke Python scripts or program files and allows access to the Python interpreter interface.

To invoke a Python script or program file use the command **python <python_file>** and replace **<python_file>** with the relative or absolute path to the Python script or program file.

```
root@Teltonika-RUTXXX:~# python /test_py.py
Hello world!
```

Alternatively, using just the command **python** will let you enter the interpreter and write and

execute your code there.

```
root@Teltonika-RUTXXX:~# python
Python 3.9.7 (default, Mar 23 2023, 08:32:35)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello world!")
Hello world!
>>>
```

For more information on how to use **python** command use the command **python --help** in the device's CLI and for how to use Python interpreter use the command **help()** in the Python interpreter interface.

```
root@Teltonika-RUTXXX:~# python --help

root@Teltonika-RUTXXX:~# python
Python 3.9.7 (default, Mar 23 2023, 08:32:35)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> help()
```

Welcome to Python 3.9's help utility!

If this is your first time using Python, you should definitely check out the tutorial on the Internet at <https://docs.python.org/3.9/tutorial/>.

Enter the name of any module, keyword, or topic to get help on writing Python programs and using Python modules. To quit this help utility and return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type "modules", "keywords", "symbols", or "topics". Each module also comes with a one-line summary of what it does; to list the modules whose name or summary contain a given string such as "spam", type "modules spam".

help>

To exit the Python interpreter interface and return to device's CLI use the command **exit()** or **quit()**.

```
root@Teltonika-RUTXXX:~# python
Python 3.9.7 (default, Mar 23 2023, 08:32:35)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
root@Teltonika-RUTXXX:~#

root@Teltonika-RUTXXX:~# python
Python 3.9.7 (default, Mar 23 2023, 08:32:35)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> quit()
```

root@Teltonika-RUTXXX:~#

Python3 Modules

abc	codeop	fcntl	io	pathlib	resource	struct	unicodedata
aifc	collections	filecmp	ipaddress	pdb	rlcompleter	subprocess	unittest
antigravity	colorsys	fileinput	itertools	pickle	runpy	sunau	urllib
argparse	compileall	fnmatch	json	pickletools	sched	symbol	uu
array	concurrent	formatter	keyword	pipes	secrets	symtable	uuid
ast	configparser	fractions	linecache	pkgutil	select	sys	venv
asynchat	contextlib	ftplib	locale	platform	selectors	sysconfig	warnings
asyncio	contextvars	functools	logging	plistlib	shelve	syslog	wave
asyncore	copy	gc	mailbox	poplib	shlex	tabnanny	weakref
atexit	copyreg	genericpath	mailcap	posix	shutil	tarfile	wsgiref
audioop	crypt	getopt	marshal	posixpath	signal	telnetlib	xdrlib
base64	csv	getpass	math	pprint	site	tempfile	xml
bdb	ctypes	gettext	mimetypes	profile	smtplib	termios	xmlrpc
binascii	curses	glob	mmap	pstats	smtplib	textwrap	xxlimited
binhex	dataclasses	graphlib	modulefinder	pty	sndhdr	this	xxsubtype
bisect	datetime	grp	multiprocessing	pwd	socket	threading	zipapp
builtins	dbm	gzip	netrc	py_compile	socketserver	time	zipfile
bz2	decimal	hashlib	ntplib	pyclbr	spwd	timeit	zipimport
cProfile	difflib	heapq	ntpath	pydoc	sqlite3	token	zlib
calendar	dis	hmac	nturl2path	pydoc_data	sre_compile	tokenize	zoneinfo
cgi	distutils	html	numbers	pyexpat	sre_constants	trace	
cgitb	doctest	http	opcode	queue	sre_parse	traceback	
chunk	email	imaplib	operator	quopri	ssl	tracemalloc	
cmath	encodings	imghdr	optparse	random	stat	tty	
cmd	enum	imp	os	re	statistics	turtle	
code	errno	importlib	ossaudiodev	readline	string	types	
codecs	faulthandler	inspect	parser	replib	stringprep	typing	