

QuecPython Basic Operation Guide

Version: 1.0.0

Date: 2020-11-13

Status: Preliminary



Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.

Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China

Tel: +86 21 5108 6236

Email: info@quectel.com

Or our local office. For more information, please visit:

<http://www.quectel.com/support/sales.htm>.

For technical support, or to report documentation errors, please visit:

<http://www.quectel.com/support/technical.htm>

Or email to support@quectel.com.

General Notes

Quectel offers the information as a service to its customers. The information provided is based upon customers' requirements. Quectel makes every effort to ensure the quality of the information it makes available. Quectel does not make any warranty as to the information contained herein, and does not accept any liability for any injury, loss or damage of any kind incurred by use of or reliance upon the information. All information supplied herein is subject to change without prior notice.

Disclaimer

While Quectel has made efforts to ensure that the functions and features under development are free from errors, it is possible that these functions and features could contain errors, inaccuracies and omissions. Unless otherwise provided by valid agreement, Quectel makes no warranties of any kind, implied or express, with respect to the use of features and functions under development. To the maximum extent permitted by law, Quectel excludes all liability for any loss or damage suffered in connection with the use of the functions and features under development, regardless of whether such loss or damage may have been foreseeable.

Duty of Confidentiality

The Receiving Party shall keep confidential all documentation and information provided by Quectel, except when the specific permission has been granted by Quectel. The Receiving Party shall not access or use Quectel's documentation and information for any purpose except as expressly provided herein. Furthermore, the Receiving Party shall not disclose any of the Quectel's documentation and information to any third party without the prior written consent by Quectel. For any noncompliance to the above requirements, unauthorized use, or other illegal or malicious use of the documentation and information, Quectel will reserve the right to take legal action.

Copyright

The information contained here is proprietary technical information of Quectel Wireless Solutions Co., Ltd. Transmitting, reproducing, disseminating and editing this document as well as using the content without permission are forbidden. Offenders will be held liable for payment of damages. All rights are reserved in the event of a patent grant or registration of a utility model or design.

Copyright © Quectel Wireless Solutions Co., Ltd. 2020. All rights reserved.

About the Document

Revision History

Version	Date	Author	Description
-	2020-09-02	Baron	Creation of the document.
1.0.0	2020-11-13	Kenney	Preliminary

Contents

About the Document.....	3
Contents.....	4
1 Introduction	5
2 System Booting.....	6
3 File System	8
3.1. Download the Script.....	8
3.2. Check File	9
3.3. Command Interaction.....	10
4 Script Execution.....	11
5 Compiling Byte-code	12
6 Appendix A References.....	13

1 Introduction

This document introduces the basic operations of QuecPython, including operations on file system, the scripts execution and byte-code compilation.

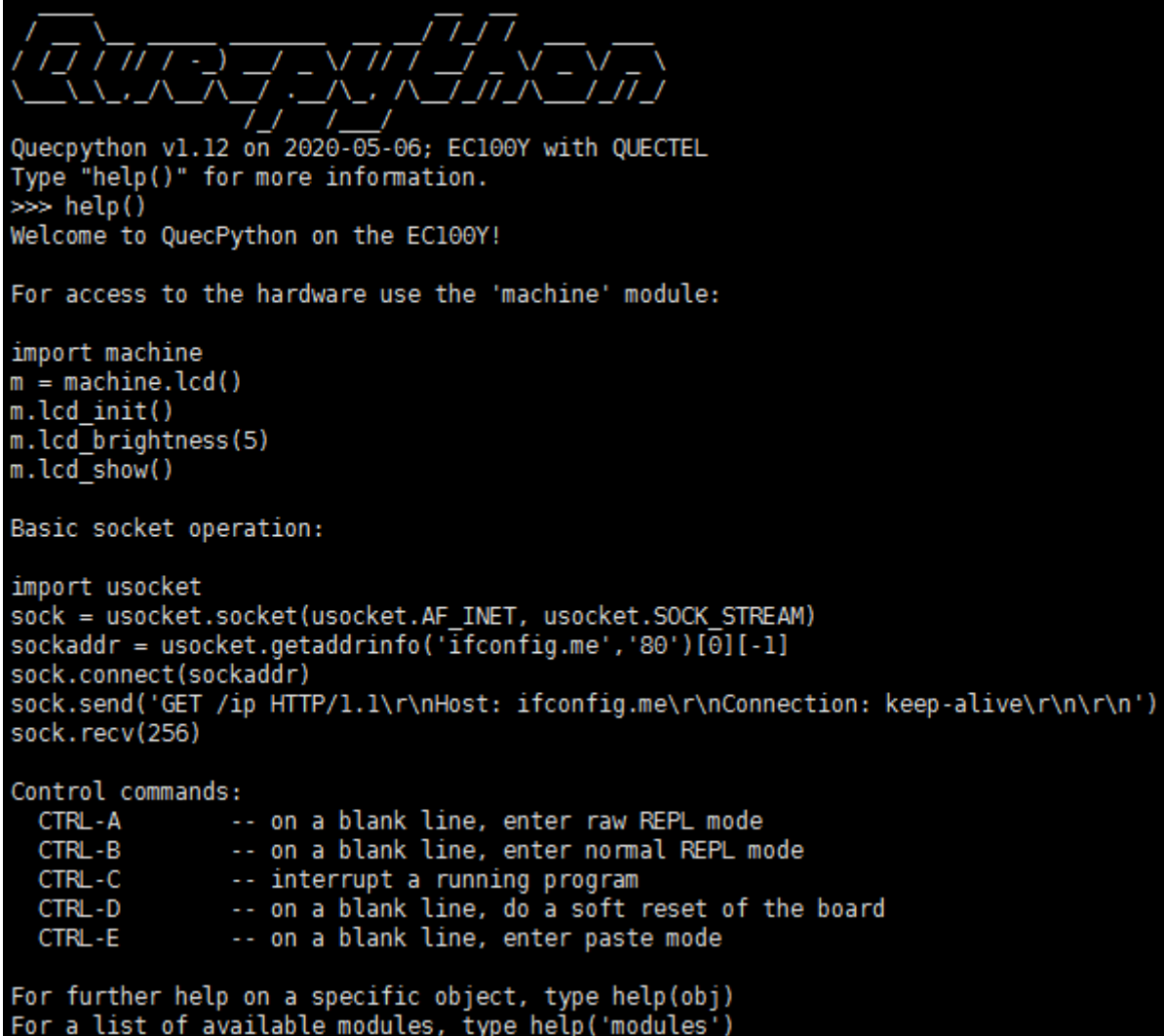
QuecPython takes the main UART as a channel for receiving command and data. All operations related to QuecPython are completed through the main UART.

The applicable modules:

- EC100Y-CN (This document takes this module as an example)
- EC600S-CN

2 System Booting

After the QuecPython booting, and the interactive interpreter that is similar to Linux shell is started on the main UART. You can execute commands and view the returned value timely through the interactive interpreter.



```
Quecpython v1.12 on 2020-05-06; EC100Y with QUECTEL
Type "help()" for more information.
>>> help()
Welcome to QuecPython on the EC100Y!

For access to the hardware use the 'machine' module:

import machine
m = machine.lcd()
m.lcd_init()
m.lcd_brightness(5)
m.lcd_show()

Basic socket operation:

import usocket
sock = usocket.socket(usocket.AF_INET, usocket.SOCK_STREAM)
sockaddr = usocket.getaddrinfo('ifconfig.me', '80')[0][-1]
sock.connect(sockaddr)
sock.send('GET /ip HTTP/1.1\r\nHost: ifconfig.me\r\nConnection: keep-alive\r\n\r\n')
sock.recv(256)

Control commands:
CTRL-A      -- on a blank line, enter raw REPL mode
CTRL-B      -- on a blank line, enter normal REPL mode
CTRL-C      -- interrupt a running program
CTRL-D      -- on a blank line, do a soft reset of the board
CTRL-E      -- on a blank line, enter paste mode

For further help on a specific object, type help(obj)
For a list of available modules, type help('modules')
```

Figure 1: Interactive Interpreter

In which,

- Execute **help(obj)** command to view the help information.
- Execute **dir(obj)** command to check the detailed method provided.
- Execute **help('modules')** command to list the class library currently supported.

When the system booting the module, not only the hardware resource is initialized, but also partitions are mounted and the initialization scripts are executed automatically.

The initialization scripts include the following:

- *__boot.py*: Initializes resources, such as mounting partitions when booting the module. This script is embedded in the factory firmware.
- *main.py*: Initializes users. This script is executed after the system initialization is completed.

3 File System

QuecPython divides a space of 5 MB as user partition where stores the customized configurations, scripts and other files. The user partition is mounted to directory '/' automatically when the module is started.

QuecPython also provides a class library *uos* to access or operate file system, see *Quectel_QuecPython_Class_Library_Introduction*. The following code shows how to create a file, write/read the file content under the current directory.

```
import uos

# create a file
f=open('test.txt','w')
f.write('hello quecpython!\n')
f.write('123456789abcdefg!\n')
f.close()
# read a file
f=open('test.txt', 'r')
print(f.readline())
print(f.readline())
f.close()
```

For convenient operation, you can use the *QPYcom.exe* tool to perform regular file system operations.

3.1. Download the Script

Step 1: Decompress the *QPYcom.zip* under *tools* directory of SDK package, get *QPYcom.exe* and double-click to run it.

Step 2: Enter the "**Download**" Tab and click "**Creation**" button to create the download program.

Step 3: Click "+" button to select the script to be downloaded.

Step 4: Click the inverted triangle button at the bottom right of the interface, switch to "download mode".

Step 5: Click "**Download script**" button to start the download process.

Step 6: When the progress bar displays "100%", it means that the download is complete, and you can enter the "File" Tab to view the file details in the module.

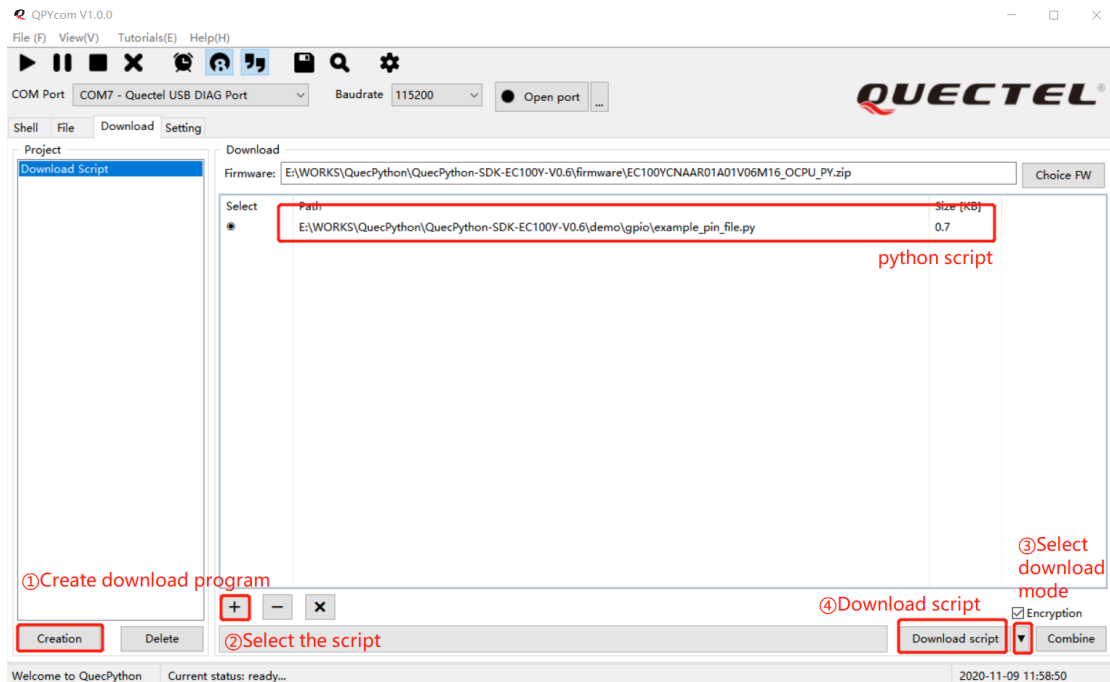


Figure 2: The Tool Interface to the Script Download

3.2. Check File

Run *QPYcom.exe* tool, press "View" --> "File browse", to perform the file uploading, viewing, adding and deleting the between the local and the module. The operation interface and involved buttons are shown in the following.

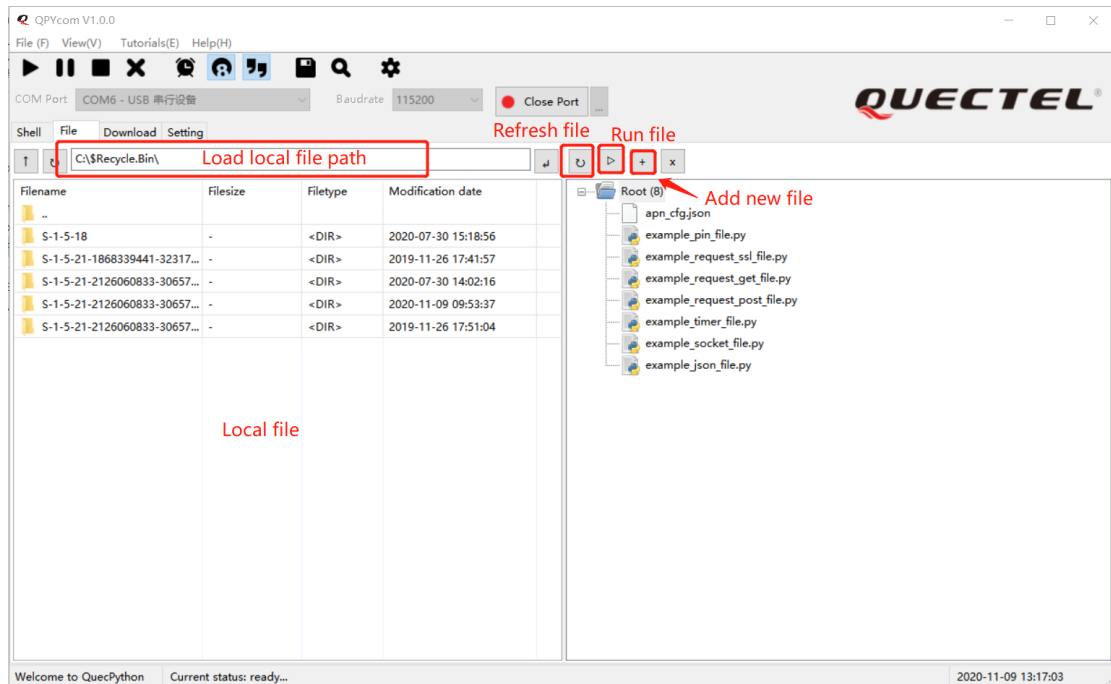


Figure 3: The File Operation between the Local and the Module

3.3. Command Interaction

Run *QPYcom.exe* tool, press "View" --> "Interactive Command Line" to enter the command interaction interface. In the interactive interface, you can manually interact with the module, and the description of the main interaction interface is shown in the figure.

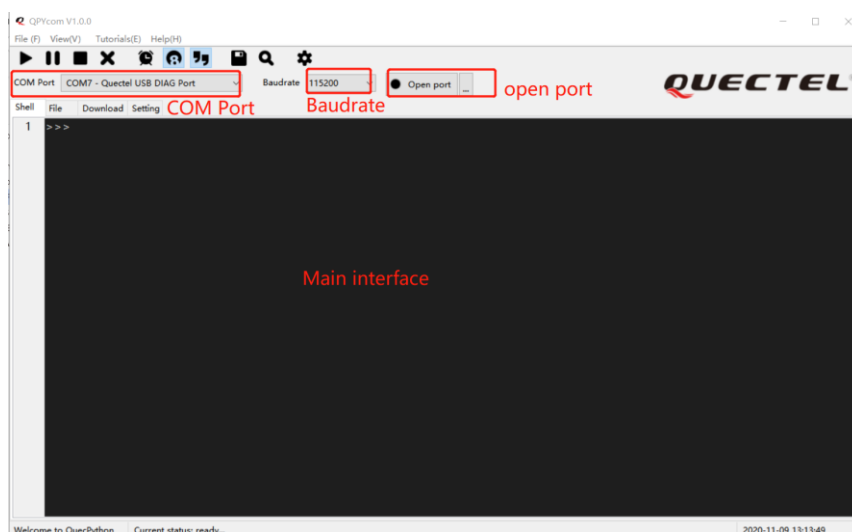


Figure 4: The Main Interaction Interface

4 Script Execution

In order to facilitate the testing locally, there is no need to upload script files to the module for execution. The *QPYcom.exe* tool provides a method to directly execute local scripts on the module side.

Step 1: Decompress the *QPYcom.zip* under *tools* directory of SDK package, get *QPYcom.exe* and double-click to run it.

Step 2: Enter "File" Tab.

Step 3: Press the button in the red frame to execute the script.

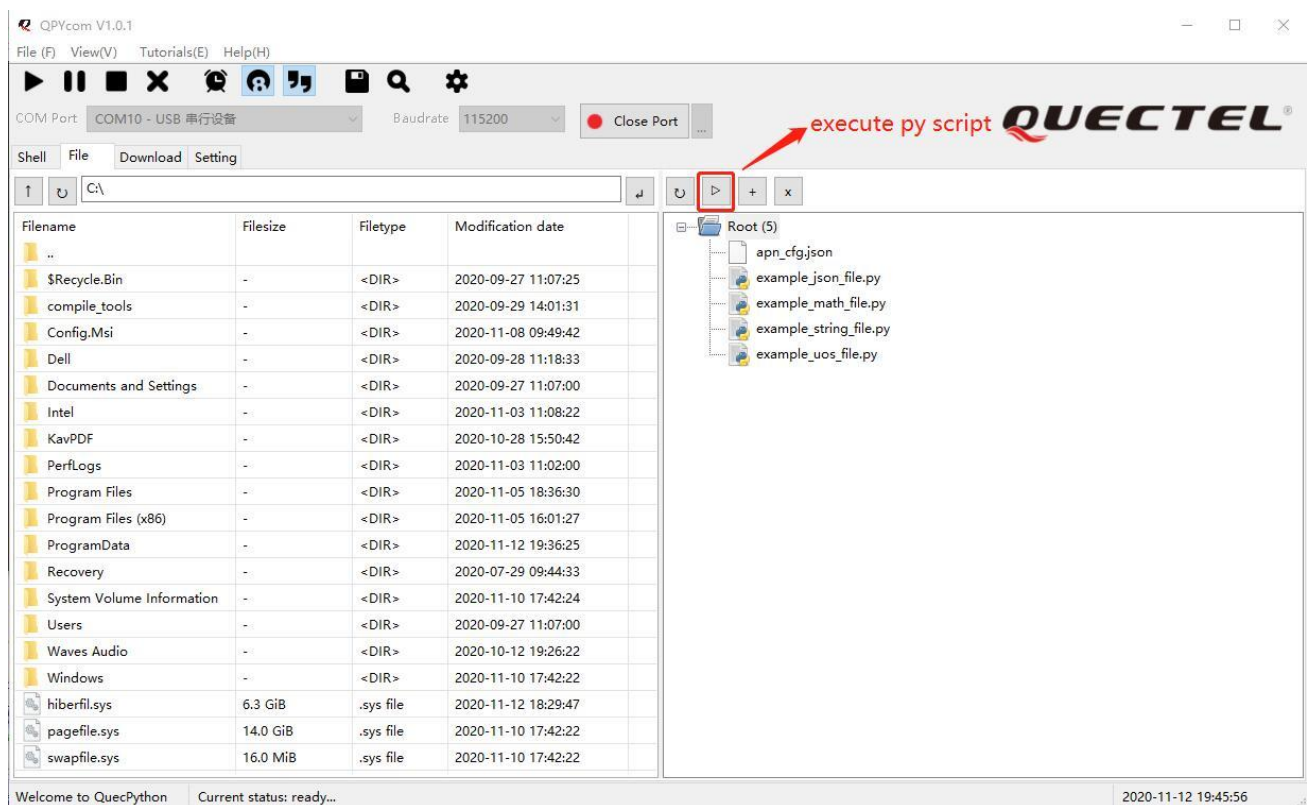


Figure 5: The Script Execution

5 Compiling Byte-code

To speed up the code execution process and improve the user code security, Quectel provides mpy-cross tool to compile the customer's *py* script to byte-code. You can embed the byte-code to the firmware or store it at the file system for other script, for more details, please refer to *Quectel_QuecPython_mpy-cross_User_Guide*.

The example of compiling *py* script to byte-code is shown as below:

```
mpy-cross.exe -o test.mpy -s test.py -march=armv7m test.py
```

6 Appendix A References

Table 1: Terms and Abbreviations

Abbreviation	Description
UART	Universal Asynchronous Receiver/Transmitter
