# Preface

### *Scope*

This document describes what to download from github, namely the required software and example files. You do not need Android Studio nor build any apk file. It further explains how to install everything up to the point of starting the emulator. Once that works you do not need this document any longer. For help on actually using the emulator see the other documents as listed in the Readme on Github. For beginners the next document is the “Introduction to the emulator”.

### *Tested system settings*

- Windows10, 64 bit, Windows11, 64 bit
- Android 9 up to 13; Android 14 possible now with QPythonPlus
- python 3.7, 3.8, 3.10, 3.12
- AAPS versions for logfile: oref1 & SMB for 2.3 – 3.3-dev
- AAPS versions for determine_basal: oref1 & SMB for 2.5 – 3.3-dev

I know other users installed it successfully on MAC or Linux.

### *Prepare Installation*

Download the Github repo (https://github.com/ga-zelle/AAPS-Emulator) as a zip archive. The next steps depend on where you want the emulator to run.

# Installation On Windows10/11

Before you start is a good idea to now think about how you organise your input and output data on your PC. You may want to create virtual drives[1] to avoid very long file path names whenever you need to provide those filenames. There are 3 groups of files involved, each with their own drive:

S: Holds the software for the emulator itself
L: holds your study topics with subfolders per project like “L:\tutorials\”
Z: holds your zipped logfiles; may be grouped in subfolders by month like “Z:\2023-01\”

Extract the zip-archive downloaded from Github to a folder containing

- "determine_basal.py" into “S:” as the emulation of the original java or Kotlin version
- "emulator_GUI.py" into “S:” which manages the user input in an interactive window
- create a shortcut on your desktop to point to this GUI file.
- "emulator_batch.py" into “S:” which alternatively manages the user input in a DOS window
- "emulator_core.py" into “S:” the main programme which manages the whole process
- "<variant_label>.vdf" into “L:” which contains the definition of the settings you want to change

To get python if you do not have it yet on your PC the best option is to download its installer from https://realpython.com/installing-python/#how-to-install-python-on-windows. When the installer starts select user specific options rather than standard configuration. In the dialogue that follows select

- check that install for all users is selected (may require admin rights at start of install)
- add python to the system variable for PATH
- select to also download and install “pip”; this makes it easier to get further required libraries like matplotlib mentioned next.

After the python installation is completed you need to download the matplotlib library. This is done in a DOS window by entering ”pip install matplotlib” on the command line.

The last action to make python fit for this project is to define a system variable[2] PYTHONUTF8=1 to allow use of special characters from the UTF-8 character set like the Greek Δ for labelling next-Δ.

### *Verify the emulator starts*

Start by clicking on the shortcut created on the desktop during the installation process. This opens a DOS window. The DOS window should remain empty but may be error messages will show up and I recommend to check for those before quitting the GUI. I tried to catch such messages in the code for display in the form itself but you never know.

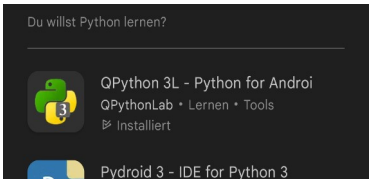
After opening that DOS window another window opens which contains a form itself to manage the whole process. The size of the window may be adjusted so longer filenames are displayed without scrolling. In general, boxes with white background are used to enter text, mostly filenames. Boxes with rounded corners and grey background are command buttons to start certain actions:

- Browse - will start the standard file selection dialogue to find the respective file
- Edit - loads that file into your standard Editor (as long as your system has assigned a standard app for that file type)
- Show - similar to „Edit“ apart from the result PDF where it uses your standard PDF viewer

The process goes from top to bottom and from left to right but of course you can jump to wherever. First, you select the working folder, i.e. the one containing your variant definition file and later the result files.

1 See appendix V how to create a virtual drive
2 See appendix S how to define a system variable

# Installation on the AAPS phone




### *For versions Android13 and below*

- From the play store download and install qpthon3 („QPython 3L – Python for Android“ by QpythonLab). In some countries like USA it is not available in the play store but can be downloaded from https://github.com/qpython-android/qpython3/releases/tag/3x-v3.0.0

- This should create a folder „qpython“ at the top level. This holds the python scripts and the final steps listed further down are equivalent for the higher Android versions.

### *For versions Android14 and above*

- Download and install the new QPythonPlus version from this Google drive location https://drive.google.com/drive/folders/1JyPawfQGLQ_41maZCU1AGjGez-74GEDT. Pick at least a 3.8 version.
- Here the QPythonPlus folder for scripts is located in the traditional place, namely in *...Android/data/org.python.plus/scripts3*. As a consequence of the much more stringent access rules for Android14 you can only save/edit/delete files there from a PC connected by USB.

  I found the USB access very tedious when updating a script file because in a first step it only deleted the old version, but then I had to cancel the USB connection, re-establish it and navigate to the folder again before being allowed to copy the new versions in. For those in the know using adb-commands might be a slicker alternative.

### *Final steps for all Android versions*

- Go to the python subfolder „scripts3“
- From the Github archive downloaded above extract the following 3 script files (i.e. the python programmes) to this subfolder
  - „determine_basal.py“
  - „emulator_batch.py“
  - „emulator_core.py“
- "<variant_label>.vdf”, the vdf-file is the same as on Windows and is copied to the folder containing the AAPS logfiles.
- “<your configuration>.config” is also copied to the logfile folder and contains information that does not change often. The software folder in the repo contains “5minute_emulator_std.config” as an example to get you going.

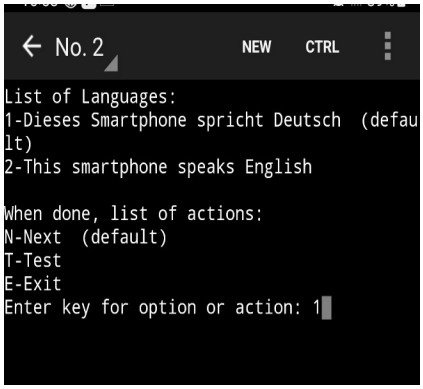
### *Verify the emulator starts on the AAPS phone*

For lower level Androids - On the phone press the *QPython3L* button created during installation. There, press *Programs*, select *emulator_batch.py* and finally select *Run*.

For higher level Androids - On the phone press the *QPython plus* button created during installation. At the top of the next screen click *Projects and Scripts*. On the next screen click *emulator_batch.py.*

Do not worry about the specifics of the upcoming dialogues. They are dealt with in a dedicated document “How to run the emulator on the phone.pdf”. Here, we just want to get through to confirm that everything is setup correctly.

The first dialogue is used to select the language for the speech synthesis.




Select “2”, thereafter select ENTER if you want to switch to English.

Select „Test“ to listen to a sample speech synthesis.

Select „Next“ to proceed.

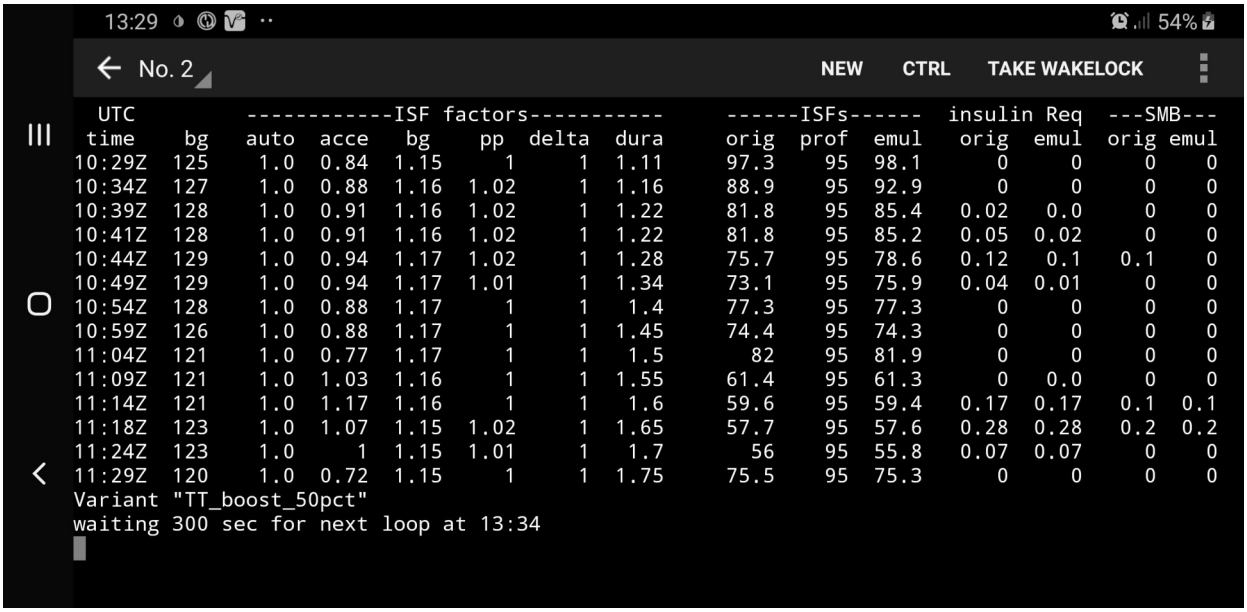
The next dialogue is for selecting your variant definition file. Use a “noChange” version or the one that comes with the examples. All vdf-files found in the logfile folder will be listed.

The next dialogue is used to select your favourite configuration file. Use one of the 5-minute standard files that came with the installation.

After you activate the selection you get a message of how many columns the selected tabular output will occupy. This is a convenient time to rotate the phone to landscape to prepare for the result table display.

Press ENTER-key to proceed.

Now you should see a screen something like this:




If so, congratulations! You are done with setting up your phone for the emulator. Next you can read the “Introduction to the AAPS emulator” for first steps on how to use it.

Status: 20-JUL-2025 @ 19:00

# Appendix S How to define a new system variable

You can search the web for tips. Here is an example instruction set taken from

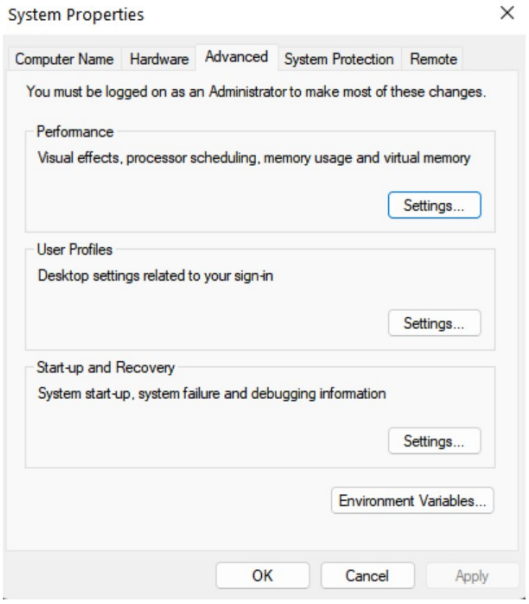
https://geekflare.com/system-environment-variables-in-windows/

**Steps for Windows 10**

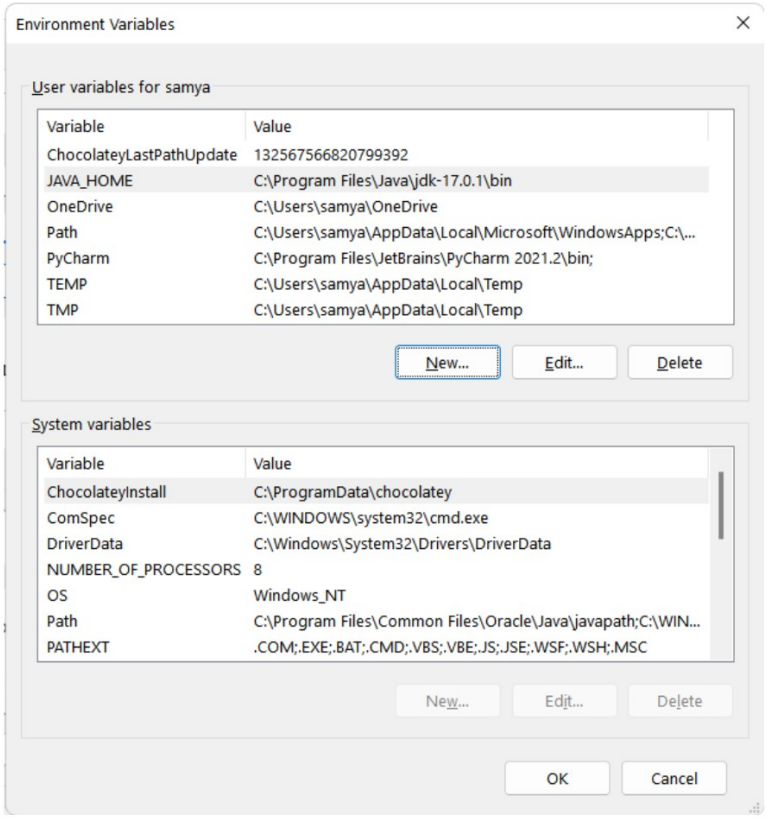
Go to settings and enter the "About" menu.

Now go to "Advanced system settings." The System Properties dialogue box should appear on your screen.

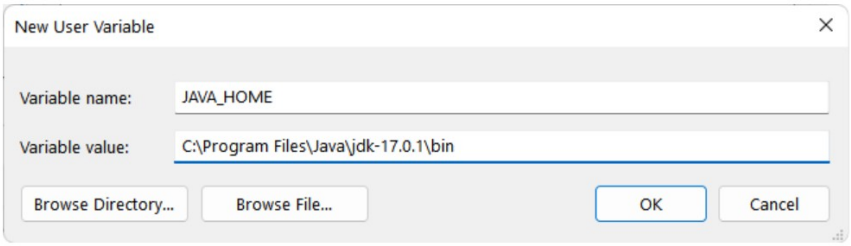
Click on the "Advanced" tab and select "Environment Variables." Windows 10 will now display the entire list of user and system variables stored on your computer.




Click on "Environment Variables" to get:



Click “New…” underneath the System variables. A form opens where you enter PYTHONUTF8 as name and 1 as value.




Click “OK” a few times to finish the assignment and close the forms.

# Appendix V: How to create a virtual drive