Setting Up Your QMK Environment

Setting Up Your QMK Environment

Before you can build keymaps, you need to install some software and set up your build environment. This only has to be done once no matter how many keyboards you plan to compile firmware for.

1. Prerequisites

There are a few pieces of software you'll need to get started.

- Text editor
  - You'll need a program that can edit and save plain text files. The default editor that comes with many OS's does not save plain text files, so you'll need to make sure that whatever editor you chose does.
- Toolbox (optional)
  - A graphical program for Windows and macOS that allows you to both program and debug your custom keyboard

If you haven't worked with the Linux/Unix command line before, there are a few basic concepts and commands you should learn. These resources will teach you enough to be able to work with QMK.

2. Prepare Your Build Environment

We've tried to make QMK as easy to set up as possible. You only have to prepare your Linux or Unix environment, then let QMK install the rest.

QMK maintains a Bundle of MSYS2, the CLI and all necessary dependencies. It also provides a handy `QMK MSYS` terminal shortcut to boot you directly into the correct environment.

Prerequisites

You will need to install QMK MSYS. The latest release is available here.

▼ Advanced Users

This process is not recommended for new users.

If you'd like to manually install MSYS2, the following sections will walk you through the process.

Prerequisites

You will need to install MSYS2. Once installed, close any open MSYS terminals (purple icon) and open a new MinGW 64-bit terminal (blue icon) from the Start Menu.

NOTE: The MinGW 64-bit terminal is *not* the same as the MSYS terminal that opens when installation is completed. Your prompt should say "MINGW64" in purple text, rather than "MSYS". See this page for more information on the differences.

Installation

Install the QMK CLI by running:

```
pacman --needed --noconfirm --disable-download-timeout -S git mingw-w64-
x86_64-python-qmk
```

QMK maintains a Homebrew tap and formula which will automatically install the CLI and all necessary dependencies.

Prerequisites

You will need to install Homebrew. Follow the instructions on https://brew.sh.

If you are using an Apple Silicon machine, the installation process will take significantly longer because GitHub actions do not have native runners to build binary packages for the ARM and AVR toolchains.

Installation

Install the QMK CLI by running:

```
brew install qmk/qmk/qmk
```

Note for WSL users: By default, the installation process will clone the QMK repository into your WSL home directory, but if you have cloned manually, ensure that it is located inside the WSL instance instead of the Windows filesystem (ie. not in `/mnt`), as accessing it is currently extremely slow.

Prerequisites

You will need to install Git and Python. It's very likely that you already have both, but if not, one of the following commands should install them:

- Debian / Ubuntu / Devuan: `sudo apt install -y git python3-pip`
- Fedora / Red Hat / CentOS: `sudo yum -y install git python3-pip`
- Arch / Manjaro: `sudo pacman --needed --noconfirm -S git python-pip libffi`
- Void: `sudo xbps-install -y git python3-pip`
- Solus: `sudo eopkg -y install git python3`
- Sabayon: `sudo equo install dev-vcs/git dev-python/pip`
- Gentoo: `sudo emerge dev-vcs/git dev-python/pip`

Installation

Install the QMK CLI by running:

```
python3 -m pip install --user qmk
```

Community Packages

These packages are maintained by community members, so may not be up to date or completely functional. If you encounter problems, please report them to their respective maintainers.

On Arch-based distros you can install the CLI from the official repositories (NOTE: at the time of writing this package marks some dependencies as optional that should not be):

```
sudo pacman -S qmk
```

You can also try the `qmk-git` package from AUR:

```
yay -S qmk-git
```

Installation

Install the FreeBSD package for QMK CLI by running:

```
pkg install -g "py*-qmk"
```

NOTE: remember to follow the instructions printed at the end of installation (use `pkg info -Dg "py*-qmk"` to show them again).

3. Run QMK Setup

Open QMK MSYS and run the following command:

```
qmk setup
```

In most situations you will want to answer `y` to all of the prompts.

Open Terminal and run the following command:

```
qmk setup
```

In most situations you will want to answer `y` to all of the prompts.

Open your preferred terminal app and run the following command:

```
qmk setup
```

In most situations you will want to answer `y` to all of the prompts.

Note on Debian, Ubuntu and their derivatives: It's possible, that you will get an error saying something like: `bash: qmk: command not found`. This is due to a bug Debian introduced with their Bash 4.4 release, which removed `$HOME/.local/bin` from the PATH. This bug was later fixed on Debian and Ubuntu. Sadly, Ubuntu reintroduced this bug and is yet to fix it. Luckily, the fix is easy. Run this as your user: `echo 'PATH="$HOME/.local/bin:$PATH"' >> $HOME/.bashrc && source $HOME/.bashrc`

Open your preferred terminal app and run the following command:

```
qmk setup
```

In most situations you will want to answer `y` to all of the prompts.

The qmk home folder can be specified at setup with `qmk setup -H <path>`, and modified afterwards using the cli configuration and the variable `user.qmk_home`. For all available options run `qmk setup --help`.

If you already know how to use GitHub, we recommend that you follow these instructions and use `qmk setup <github_username>/qmk_firmware` to clone your personal fork. If you don't know what that means you can safely ignore this message.

4. Test Your Build Environment

Now that your QMK build environment is set up, you can build a firmware for your keyboard. Start by trying to build the keyboard's default keymap. You should be able to do that with a command in this format:

```
qmk compile -kb <keyboard> -km default
```

For example, to build a firmware for a Clueboard 66% you would use:

```
qmk compile -kb clueboard/66/rev3 -km default
```

The keyboard option is the path relative to the keyboard directory, the above example would be found in `qmk_firmware/keyboards/clueboard/66/rev3`. If you’re unsure you can view a full list of supported keyboards with `qmk list-keyboards`.

When it is done you should have a lot of output that ends similar to this:

```
Linking: .build/clueboard_66_rev3_default.elf
[OK]
Creating load file for flashing: .build/clueboard_66_rev3_default.hex
[OK]
Copying clueboard_66_rev3_default.hex to qmk_firmware folder
[OK]
Checking file size of clueboard_66_rev3_default.hex
[OK]
 * The firmware size is fine - 26356/28672 (2316 bytes free)
```

Creating Your Keymap

You are now ready to create your own personal keymap! Move on to Building Your First Firmware for that.

✏️ Edit this page