Contributing Guide

==================

Thanks for your interest in contributing to this project! This document aims to

serve as a friendly guide for making your first contribution.

High-level Etcher overview

--------------------------

Make sure you checkout our [ARCHITECTURE.md][ARCHITECTURE] guide, which aims to

explain how all the pieces fit together.

Developing

----------

### Prerequisites

#### Common

- [NodeJS](https://nodejs.org) (at least v6.11)

- [Python 2.7](https://www.python.org)

- [jq](https://stedolan.github.io/jq/)

- [curl](https://curl.haxx.se/)

- [npm](https://www.npmjs.com/) (version 6.7)

```sh

pip install -r requirements.txt

```

You might need to run this with `sudo` or administrator permissions.

#### Windows

- [NSIS v2.51](http://nsis.sourceforge.net/Main\_Page) (v3.x won't work)

- Either one of the following:

- [Visual C++ 2015 Build Tools](http://landinghub.visualstudio.com/visual-cpp-build-tools) containing standalone compilers, libraries and scripts

- Install the [windows-build-tools](https://github.com/felixrieseberg/windows-build-tools) via npm with `npm install --global windows-build-tools`

- [Visual Studio Community 2015](https://www.microsoft.com/en-us/download/details.aspx?id=48146) (free) (other editions, like Professional and Enterprise, should work too)

\*\*NOTE:\*\* Visual Studio 2015 doesn't install C++ by default. You have to rerun the

setup, select "Modify" and then check `Visual C++ -> Common Tools for Visual

C++ 2015` (see http://stackoverflow.com/a/31955339)

- [MinGW](http://www.mingw.org)

You might need to `npm config set msvs\_version 2015` for node-gyp to correctly detect

the version of Visual Studio you're using (in this example VS2015).

The following MinGW packages are required:

- `msys-make`

- `msys-unzip`

- `msys-zip`

- `msys-bash`

- `msys-coreutils`

#### macOS

- [Xcode](https://developer.apple.com/xcode/)

It's not enough to have [Xcode Command Line Tools] installed. Xcode must be installed

as well.

#### Linux

- `libudev-dev` for libusb (install with `sudo apt install libudev-dev` for example)

### Cloning the project

```sh

git clone --recursive https://github.com/balena-io/etcher

cd etcher

```

### Installing npm dependencies

\*\*NOTE:\*\* Please make use of the following command to install npm dependencies rather

than simply running `npm install` given that we need to do extra configuration

to make sure native dependencies are correctly compiled for Electron, otherwise

the application might not run successfully.

If you're on Windows, \*\*run the command from the \_Developer Command Prompt for

VS2015\_\*\*, to ensure all Visual Studio command utilities are available in the

`%PATH%`.

```sh

make electron-develop

```

### Running the application

#### GUI

```sh

# Build the GUI

make webpack

# Start Electron

npm start

```

Testing

-------

To run the test suite, run the following command:

```sh

npm test

```

Given the nature of this application, not everything can be unit tested. For

example:

- The writing operating on real raw devices.

- Platform inconsistencies.

- Style changes.

- Artwork.

We encourage our contributors to test the application on as many operating

systems as they can before sending a pull request.

\*The test suite is run automatically by CI servers when you send a pull

request.\*

We also rely on various `make` targets to perform some common tasks:

- `make lint`: Run the linter.

- `make sass`: Compile SCSS files.

We make use of [EditorConfig] to communicate indentation, line endings and

other text editing default. We encourage you to install the relevant plugin in

your text editor of choice to avoid having to fix any issues during the review

process.

Updating a dependency

---------------------

Given we use [npm shrinkwrap][shrinkwrap], we have to take extra steps to make

sure the `npm-shrinkwrap.json` file gets updated correctly when we update a

dependency.

Use the following steps to ensure everything goes flawlessly:

- Run `make electron-develop` to ensure you don't have extraneous dependencies

you might have brought during development, or you are running older

dependencies because you come from another branch or reference.

- Install the new version of the dependency. For example: `npm install --save

<package>@<version>`. This will update the `npm-shrinkwrap.json` file.

- Commit \*both\* `package.json` and `npm-shrinkwrap.json`.

Diffing Binaries

----------------

Binary files are tagged as "binary" in the `.gitattributes` file, but also have

a `diff=hex` tag, which allows you to see hexdump-style diffs for binaries,

if you add the following to either your global or repository-local git config:

```sh

$ git config diff.hex.textconv hexdump

$ git config diff.hex.binary true

```

And global, respectively:

```sh

$ git config --global diff.hex.textconv hexdump

$ git config --global diff.hex.binary true

```

If you don't have `hexdump` available on your platform,

you can try [hxd], which is also a bit faster.

Commit Guidelines

-----------------

See [COMMIT-GUIDELINES.md][COMMIT-GUIDELINES] for a thorough guide on how to

write commit messages.

Sending a pull request

----------------------

When sending a pull request, consider the following guidelines:

- Write a concise commit message explaining your changes.

- If applies, write more descriptive information in the commit body.

- Mention the operating systems with the corresponding versions in which you

tested your changes.

- If your change affects the visuals of the application, consider attaching a

screenshot.

- Refer to the issue/s your pull request fixes, so they're closed automatically

when your pull request is merged.

- Write a descriptive pull request title.

- Squash commits when possible, for example, when committing review changes.

Before your pull request can be merged, the following conditions must hold:

- The linter doesn't throw any warning.

- All the tests pass.

- The coding style aligns with the project's convention.

- Your changes are confirmed to be working in recent versions of the operating

systems we support.

Don't hesitate to get in touch if you have any questions or need any help!

[ARCHITECTURE]: https://github.com/balena-io/etcher/blob/master/docs/ARCHITECTURE.md

[COMMIT-GUIDELINES]: https://github.com/balena-io/etcher/blob/master/docs/COMMIT-GUIDELINES.md

[EditorConfig]: http://editorconfig.org

[shrinkwrap]: https://docs.npmjs.com/cli/shrinkwrap

[hxd]: https://github.com/jhermsmeier/hxd

[Xcode Command Line Tools]: https://developer.apple.com/library/content/technotes/tn2339/\_index.html