# A gentle guide to Git and GitHub

## Installing third-party software

If you're thinking of using our libraries (or other third-party libraries) on your Raspberry Pi, you've probably seen references to **Git** and **GitHub**. You may be a veteran **Git** user, in which case you can skip this article, but if **Git** or **GitHub** are unfamiliar to you, read on.

Let's start by taking a step back. If you want to use someone else's' software library, there are several different ways in which you might be able to install it.

1. You might download an archive (typically a zip or tgz file). You'll usually need to unpack the archive and run some kind of installation script. That's one way you can install our **gpio-admin** and **quick2wire-python-api** libraries.
2. If you're lucky, the library might be packaged as a Debian package (let's say it's called *glockenspiel*) and stored in a repository. In that case, you'll be able to install it by invoking

```
sudo apt-get install glockenspiel
```

and the package should then be available for use.

We'll package each of our libraries like that eventually, but only when they are stable and complete. That may take a while.

## The problem of fast-changing code

At present we're changing and extending the quick2wire-python-api code quite frequently, and we'll update the **gpio-admin** library whenever we fix a bug or add a feature. If you use either of the earlier methods to install software and that software is changing fast, it's likely that your version will be out-of-date, and you may not even know it.

That's where **Git** comes in.

## Git

**Git** is a free and open source distributed version control system. Let's take that apart bit by bit.

**Git** is free and open source. You don't need to pay to use it, and the source code is freely available. In practice that means that many pairs of eyes will be scanning the code and reading any bug reports, so defects in the code are likely to be found and fixed quickly.

But what does **Git** do?

The fact that it's a version control system means that it can help you keep track of files that are frequently changed. Most version control systems maintain a history of changes and make it easy for users to compare versions, to load previous versions, and to add new versions as appropriate. Most software developers use a version control system of some sort, and many use **Git**.

**Git** has a range of features which make it attractive to developers, and one of those features is the fact that it supports *distributed* development. In other words, **Git** is designed for use by teams in which the developers work in more than one location. Most of the developers at Quick2wire have day jobs and we work on our software from home, so a distributed version control system is really useful to us.

You don't need to know about we how we use **Git** in our development process. The thing that matters is that you can use **Git**, simply and easily, to get the latest version of our software on your Raspberry Pi.

**Git** expects developers to keep all the files for a project or library together in something called a repository. That's just a grand name for a collection of files that belong together. All of our libraries are stored in **Git** repositories, and those repositories are stored on **GitHub**.

## GitHub

Hopefully that's given you a feel for **Git** and why we use it, but what about **GitHub**?

**GitHub** is a web-based service for people who want to use **Git**. It's widely used by teams who want to make some or all of their work publicly available

under an open source license. Since that's what we do, we feel that **GitHub** is a natural choice for us to store our code.

If you want to keep up-to-date with the code we've published on **GitHub**, you will need to install and use **Git**.

## Installing Git on your Pi

To install **Git**, just pen a terminal window by typing ***[Ctrl][Alt][T]*** and invoke

```
sudo apt-get install git-core
```

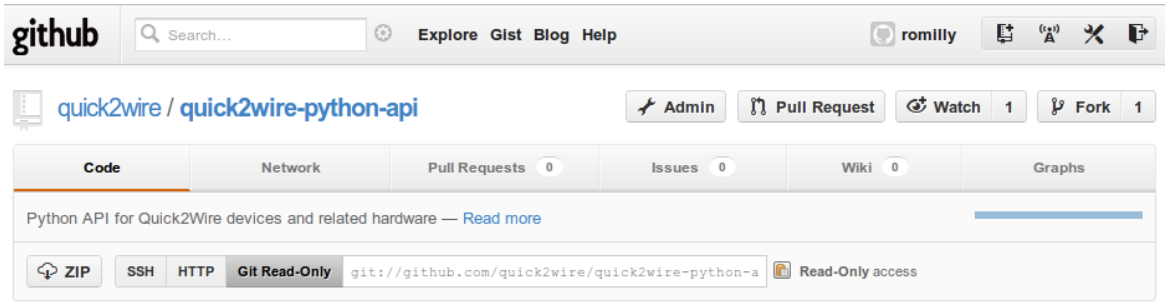
You'll be asked for your password and so long as your Pi has access to the internet and the Debian RasPi repository is on-line you'll see some activity which will show what's going on as **Git** is installed.

Once that stops you'll be ready to go.

## Using Git to download our code

When you first install a local copy of a repository using **Git**, you do so by creating what's called a *clone* of the repository.

If you take a look at the web-page for our **quick2wire-python-api** library, at the top of the web- page you'll see a panel that looks like this:




Project URI panel on **GitHub**

In order to install a local copy of the repository, click on the button marked '***Git Read-Only***' and copy the text of the URI on its right. In our case, that's ***git://github.com/quick2wire/quick2wire-python-api.git***

It's probably best to keep all your **GitHub**-based repositories in one place, so you may want to create a directory called **Git** within your home directory. (In a command window, type ***cd ~***

and then type ***mkdir git***)

When you have done so, open a command window if necessary, change to your **Git** directory by typing ***cd ~/git*** and create a copy of the **GitHub** content by typing

```
git clone git://github.com/quick2wire/quick2wire-python-api.git
```

**Git** will tell you that it's created an empty repository and will then pull down all the current files from the repository on **GitHub**.

The ***README*** file on **Git** should tell you what do to install the software in the repository. in our case the recommended approach is to add the location of your new cloned repository to your Python Path.

Now you can use the software.

But how do you keep it up to date?

## Keeping current

If you take look at the **GitHub** page for a repository you can see when the files were last changed. If you open an account at **GitHub** (which has a free account option) you can chose to *watch* one or more repositories, in which case you will get notified whenever the repository changes.

If a repository you've cloned has changed since you last updated it, it's really easy to get back up-to-date. In a command window, change to the directory in which you cloned the repository. In our case, you'd type ***cd ~/git/quick2wire-python-api*** in a command window. Now just type ***git pull*** and **Git** will update your repository so that it's an exact copy of the current version on **GitHub**.

Depending on the way type of software in the repository, you may or may not need to repeat the installation procedure. If you've updated the **gpio-admin** repository, you'll need to run through the installation process again, as described in the ***README*** for that repository in **GitHub**. In the case of the quick2wire-python-api package, you've already placed the code on your Python Path so you should be ready to use the updated library without doing anything else.

## Summary

You can use most code on **GitHub** by downloading, unpacking and installing an archive containing the code.

If the code is changing fast, and you want to stay up-to-date you may prefer to install **Git**, use the clone command to create a local copy of the repository, and use the ***git pull*** command to refresh your copy when the version on **GitHub** has changed.