#elementary OS Bug Fixing Guide

This document should contain all the steps necessary for one to propose a bug fix for one of the [elementary projects](https://launchpad.net/elementary). After reading this, you will know all it takes regarding Launchpad and Bazaar to propose bug fixes.

#Setting up Bazaar and Launchpad

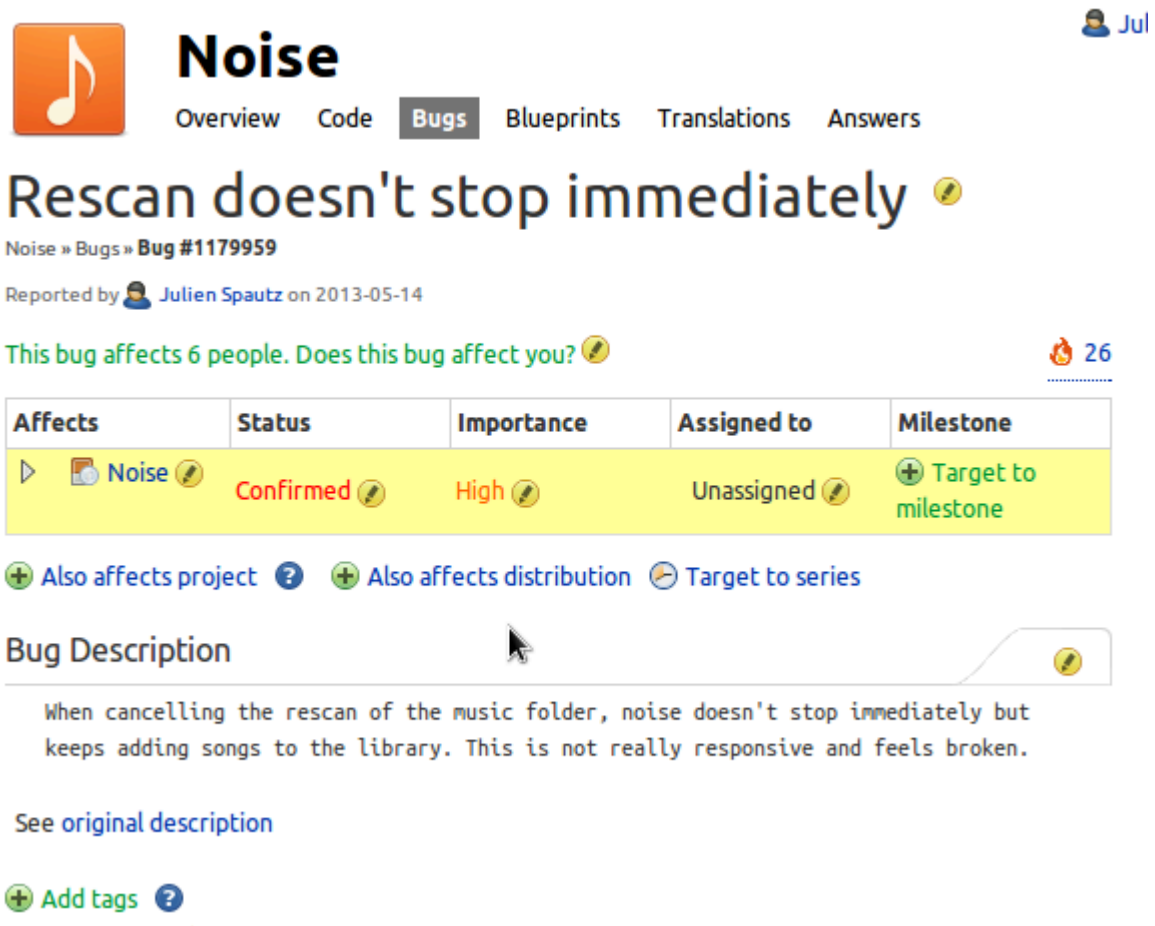
Check out [The Basic Setup](http://elementaryos.org/docs/code/the-basic-setup) for instructions.

#Picking a bug from the bug lists

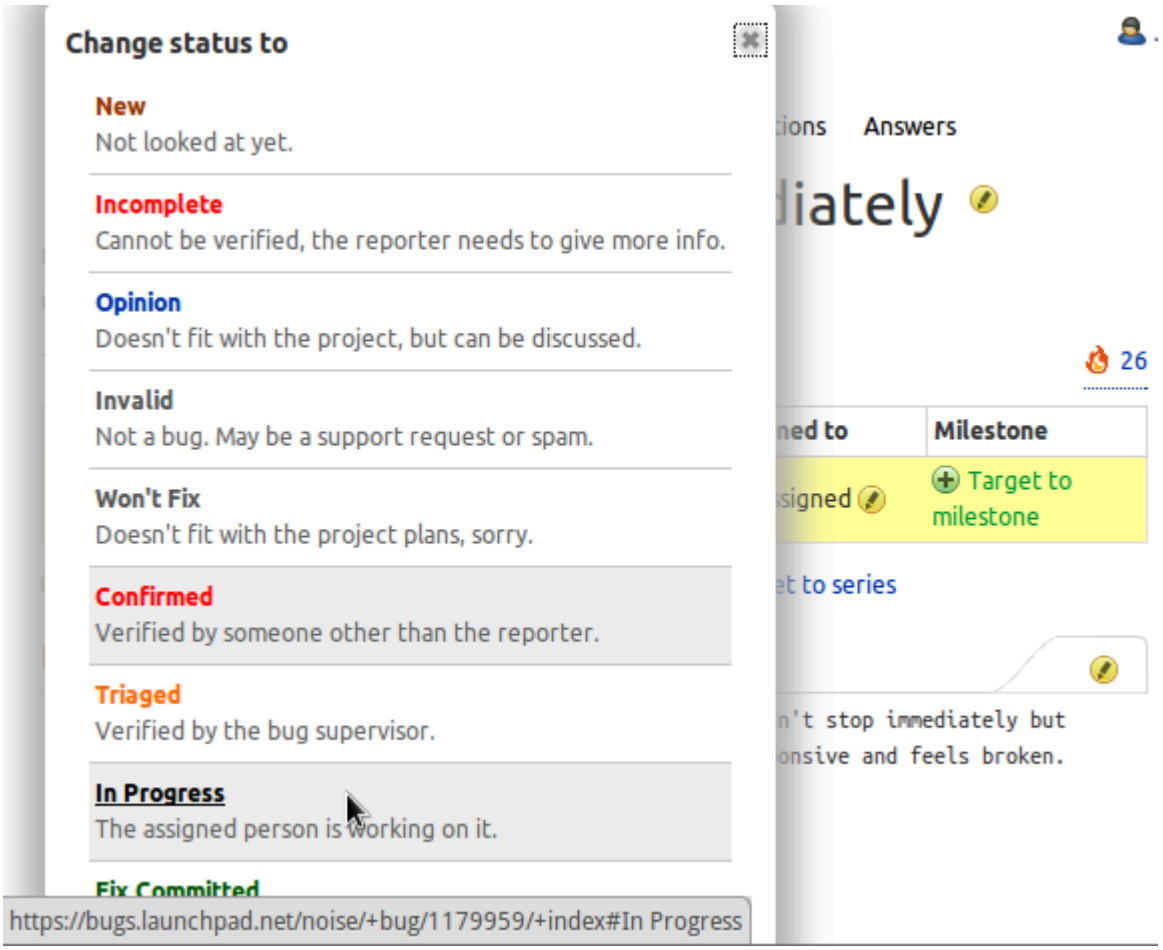
Launchpad provides lists of bugs for each one of our projects. Say, for example, that you want to check out Noise (installed under the name of Music) bugs, you simply go to <http://bugs.launchpad.net/noise> and you can find all the reported bugs and their statuses.

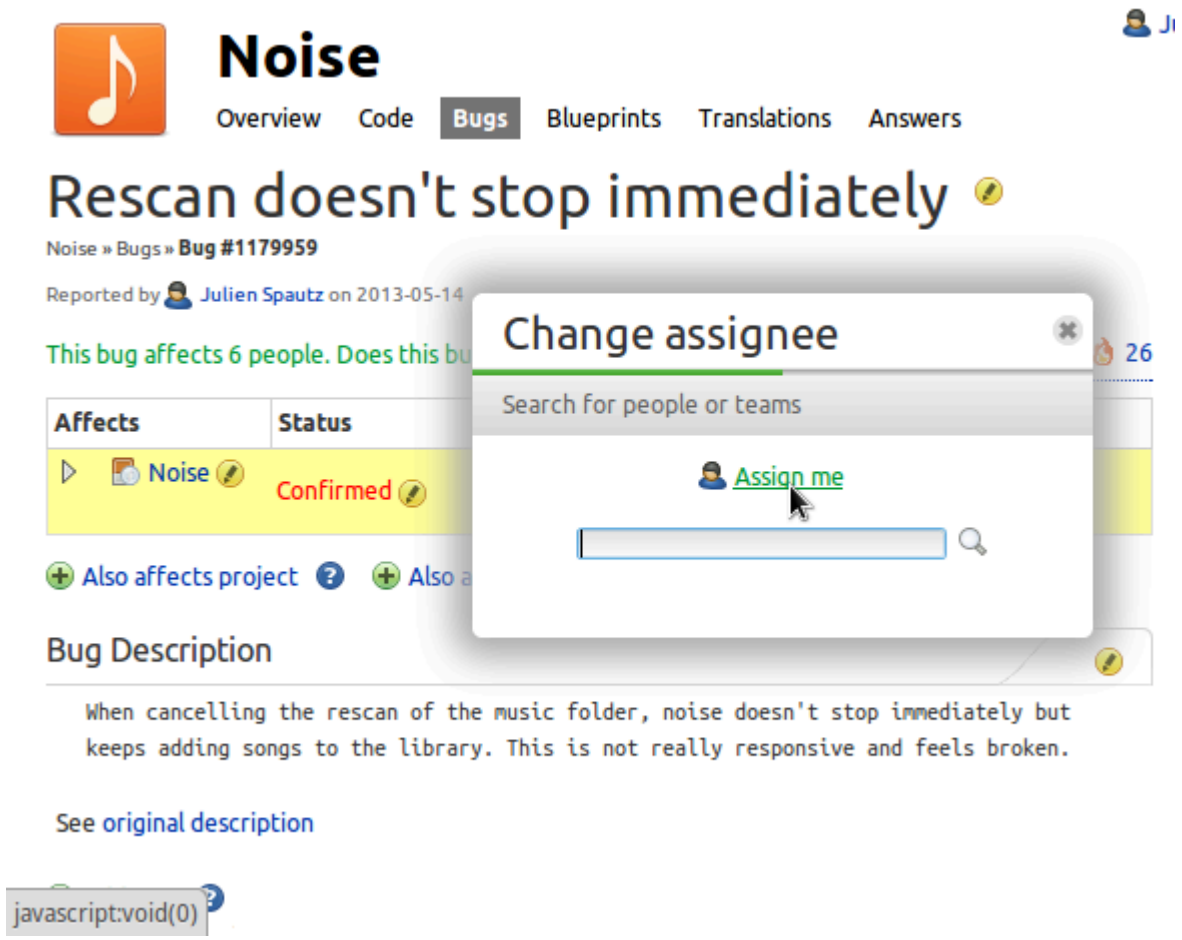
Smaller, easier bugs that are great to get started with are tagged "bitesize".

For an example, let's use the following Noise bug:



Let’s suppose you want to fix this bug. To avoid duplication of effort, when working on a bug set its status to "In Progress" and assign it to yourself (click the icon in the "Assigned to" field and then click "Assign me").





The result will look similar to this:



#Getting the source code and its dependencies

To get the source code for Noise, open your terminal and run the following:

```
bzr branch lp:noise
```

To get the necessary dependencies for building noise, run:

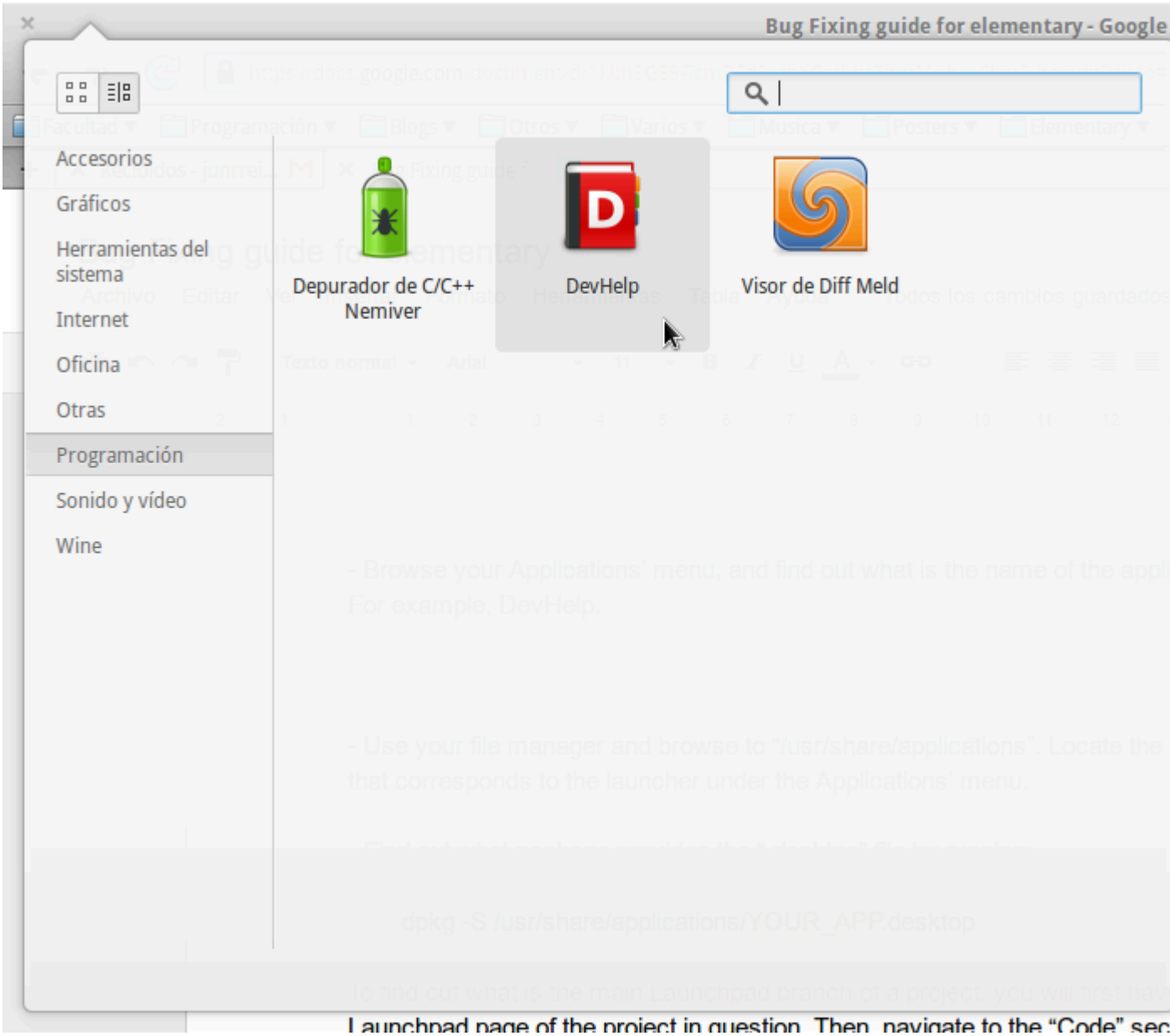
```
sudo apt-get build-dep noise
```

We need to make a clarification now. The argument for “bzr branch” is the location of a Bazaar branch (which can be online, as this case, or not), while the argument for “apt-get build-dep” is the name of the Debian package that provides the application.

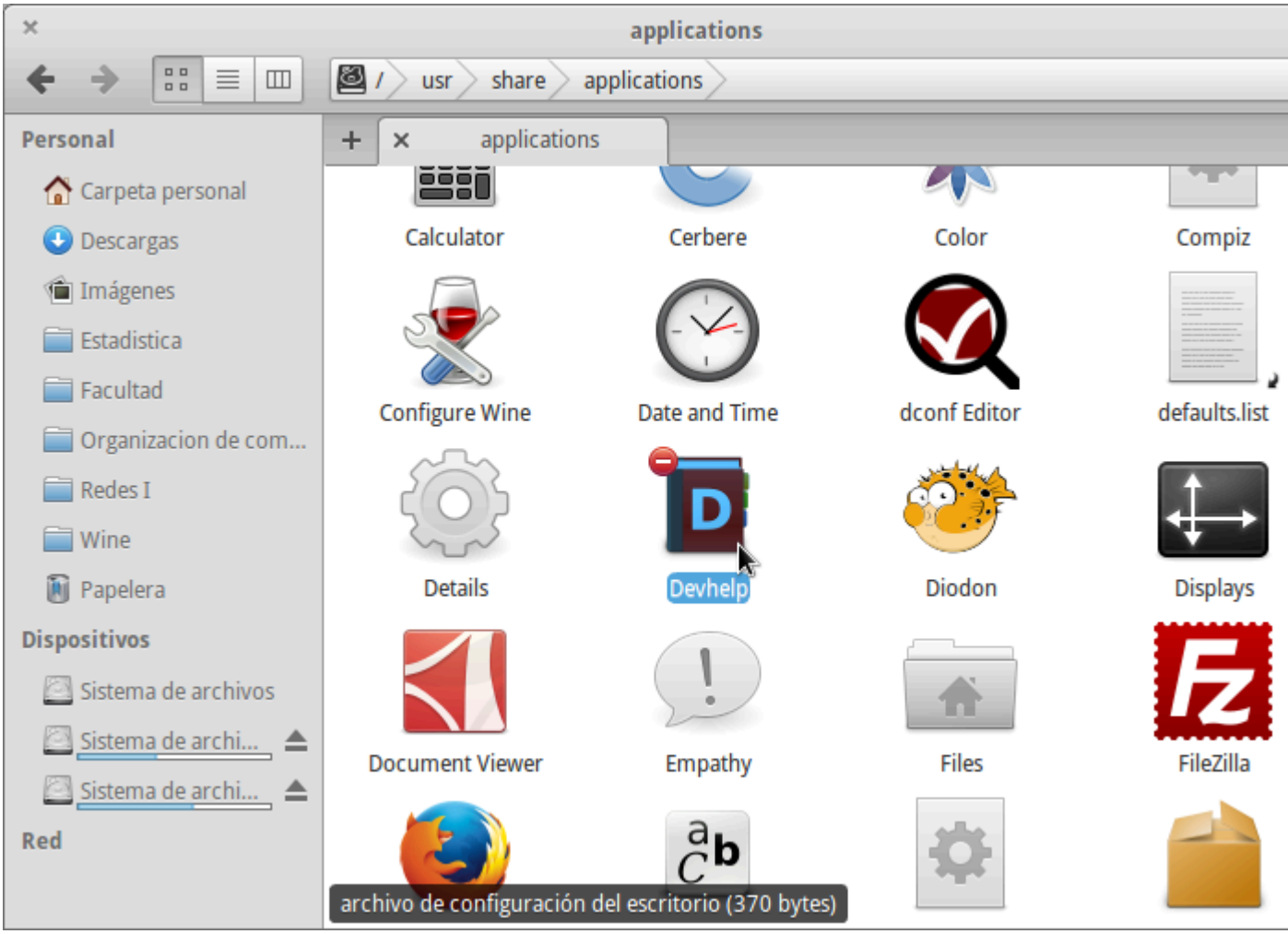
While for Noise the Launchpad branch and the Debian package have the same name, this isn’t necessarily the case for all projects. For example, Scratch has its main Launchpad branch under “lp:scratch”, but its Debian package is called “scratch-text-editor”.

To find out the name of the Debian package that provides an application, you can follow this procedure:

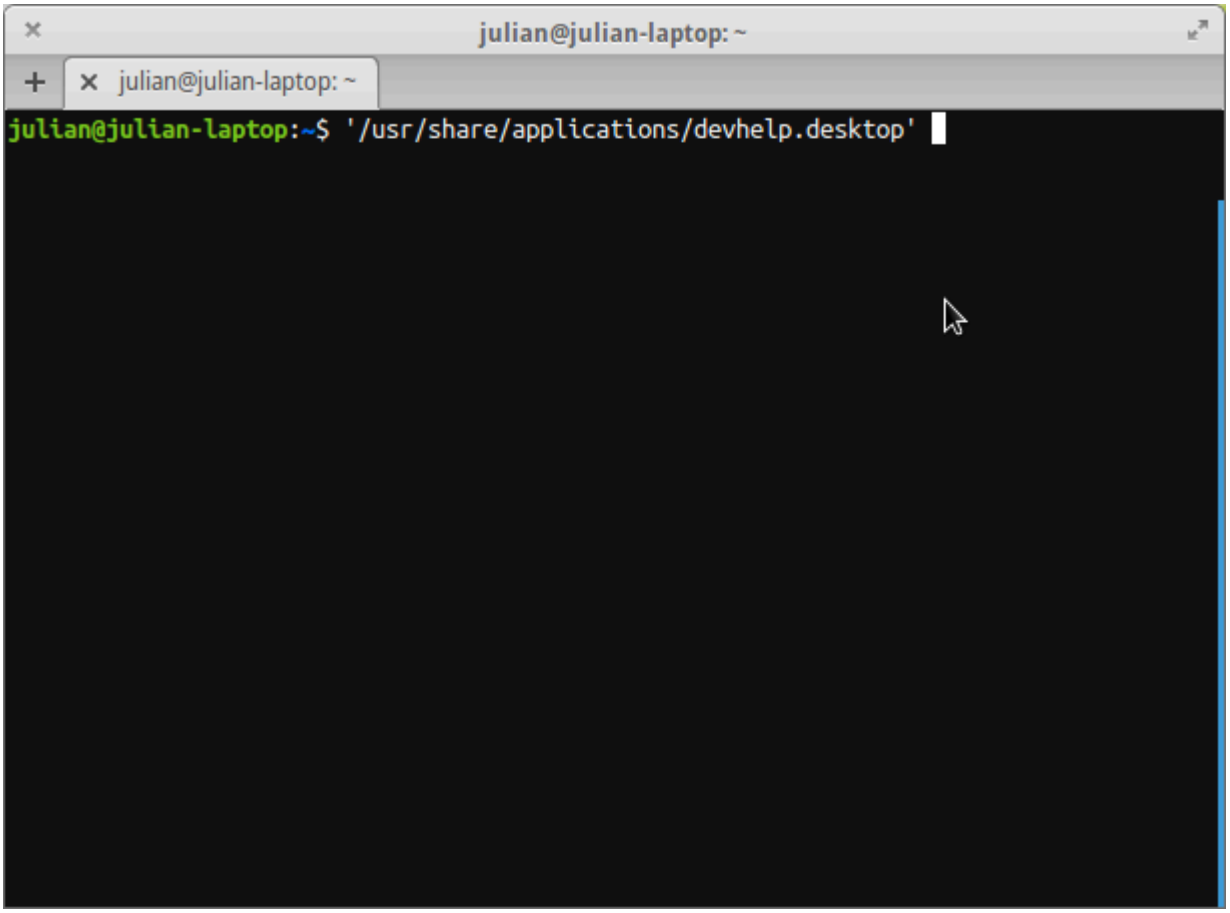
1. Browse your Applications' menu, and find out what is the name of the application launcher. For example, DevHelp.



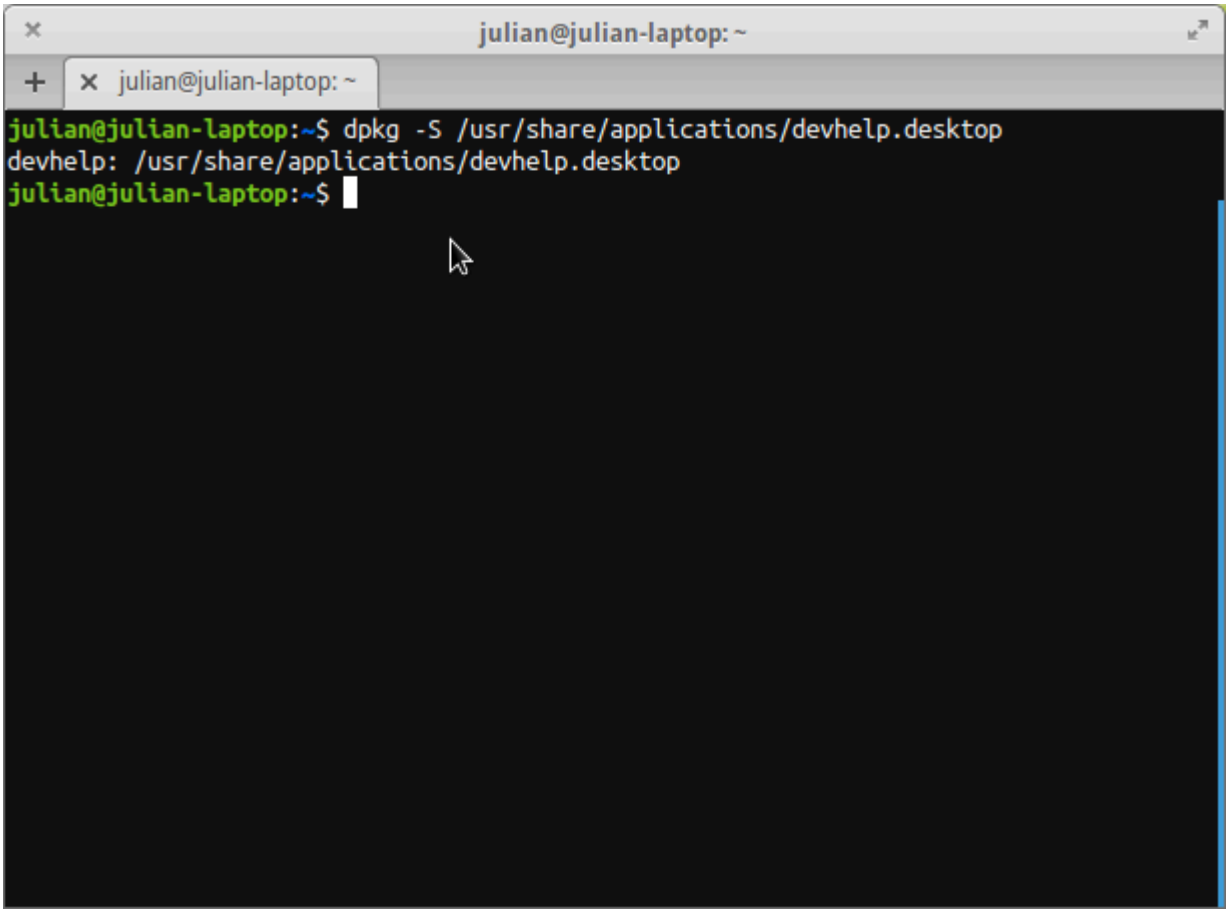
2. Use your file manager and browse to "/usr/share/applications". Locate the ".desktop" file that corresponds to the launcher under the Applications' menu.



3. Drag and drop the ".desktop" file to your terminal emulator. The path to the ".desktop" file will appear enclosed in single quotes.



4. Delete the single quotes, prepend “dpkg -S” to the file path, and run the command. That command will show which package provided that “.desktop” file, which is the package that also provided the application. In the example, the package name is “devhelp”.



To find out what is the main Launchpad branch of a project, you will first have to locate the Launchpad page of the project in question. Then, navigate to the “Code” section. At the beginning of that section, you will find the name of the main branch (also known as “development focus branch”).

You can keep the source code of this app updated by running the following in the directory containing the code:

```
bzr pull
```

#Building the application

To build Noise from its sources, use a terminal to browse to the directory of the branch you downloaded earlier, and run the following commands:

```
rm -rf build
mkdir build
cd build
cmake .. -DCMAKE_INSTALL_PREFIX=/usr
make
```

These commands will effectively build the code of any elementary project.

To test the application you have just compiled, browse to your build directory, and run the generated binary. The name (and sometimes the location) of the binary will vary from project to project. In Noise's case, run:

./src/noise

#Fixing a bug

It's a good idea to create another branch to modify for a bug fix, so you have an untouched branch which can be used for comparison:

bzr branch lp:noise name-of-your-branch

This command will create a copy of lp:noise under the directory "name-of-your-branch". This name can be whatever you want. It is a good idea to give your branch a name that holds a relation with the bug you are trying to fix. In the ongoing example, a good name for this branch would be "stop-rescanning-immediately".

If you already have a Noise branch laying around (like the one we created earlier) you can make a new branch from it without having to download everything again:

bzr branch noise/ stop-rescanning-immediately

#Committing a bug fix

After you've hacked around and managed to fix the bug, you need to commit your code using Bazaar. To do that, run the following command inside the branch directory:

bzr commit --fixes lp:*bug-number* -m "*Description of changes*"

"*bug-number*" is the name of the bug you are working on. This argument tells Launchpad to automatically link your branch with the bug you specified here.
The string given as an argument after "-m" is the commit message. A commit message is a short summary of the changes made since the previous revision, and is stored for logging purposes. Commit messages are usually written in present tense.

For our Noise bug example, a commit command will look like this:

bzr commit --fixes lp:1179959 -m "Ensure that library rescanning stops immediately when canceled to fix bug #1179959."

#Submitting a bug fix

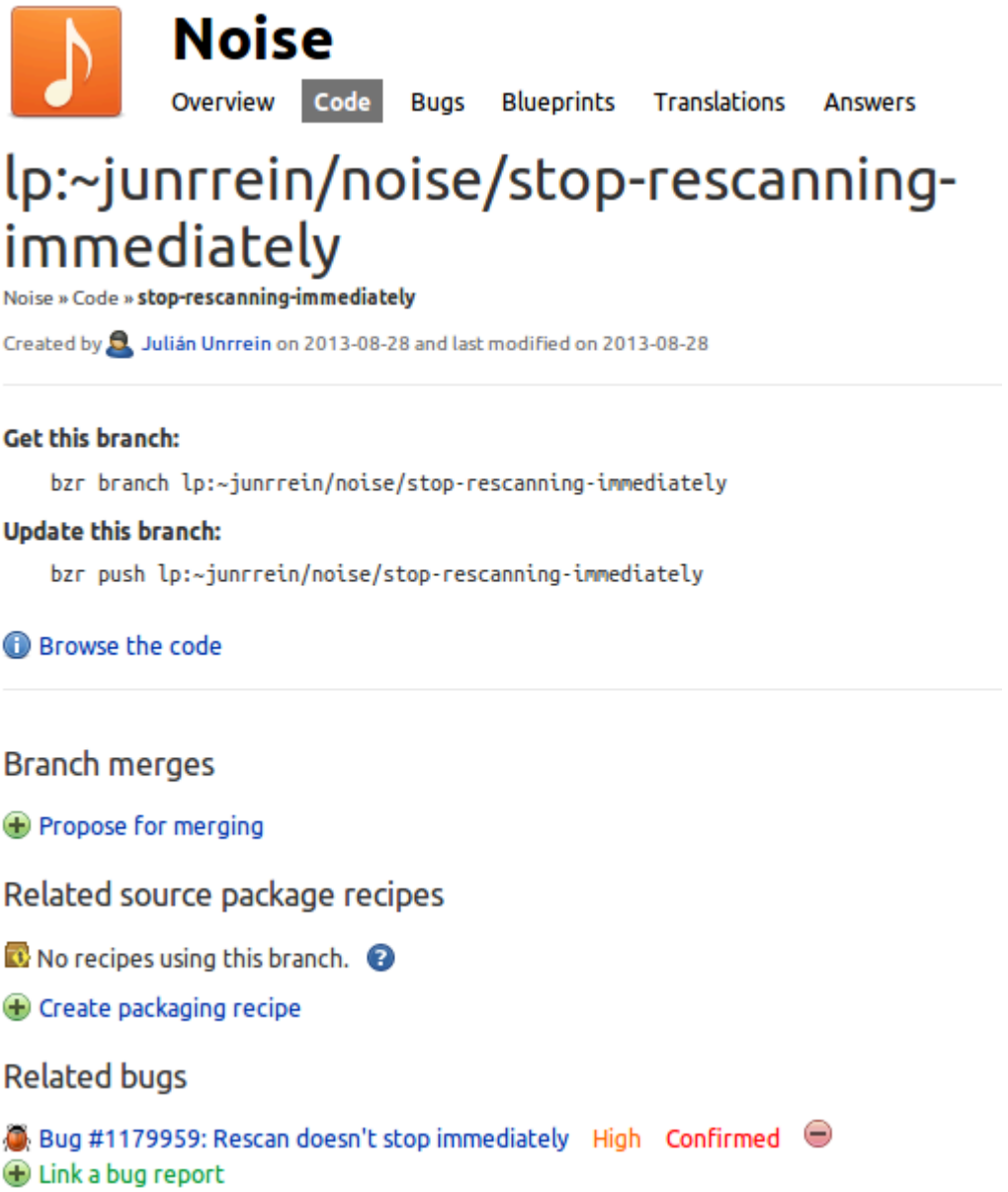
After you committed your bug fix, you will need to submit a merge proposal to get it into the main project's source code.

To do that, you will first need to upload your branch to Launchpad:

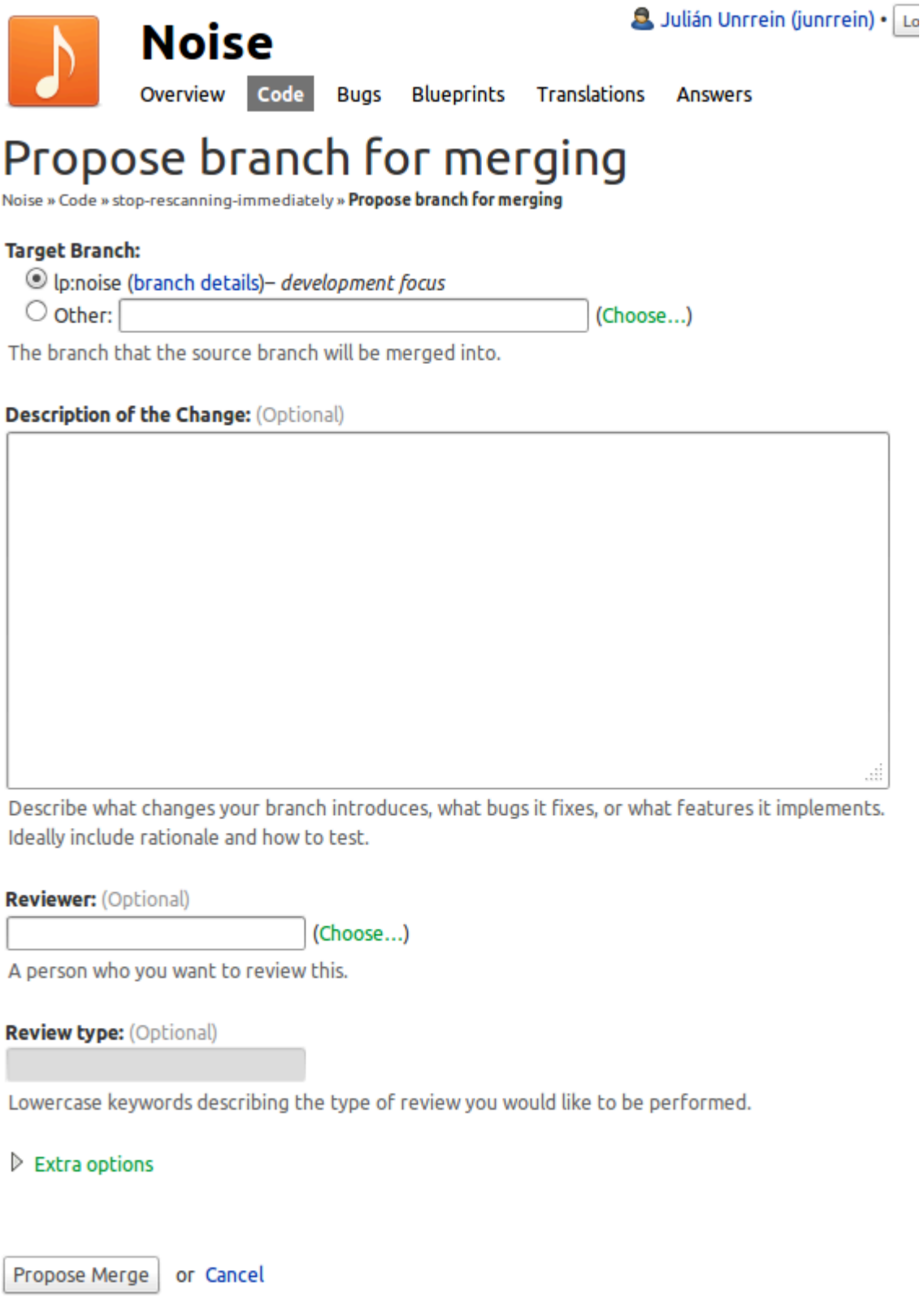
```
bzr push lp:~your_username/noise/stop-rescanning-immediately
```

The following command will open your branch summary in the Launchpad website:

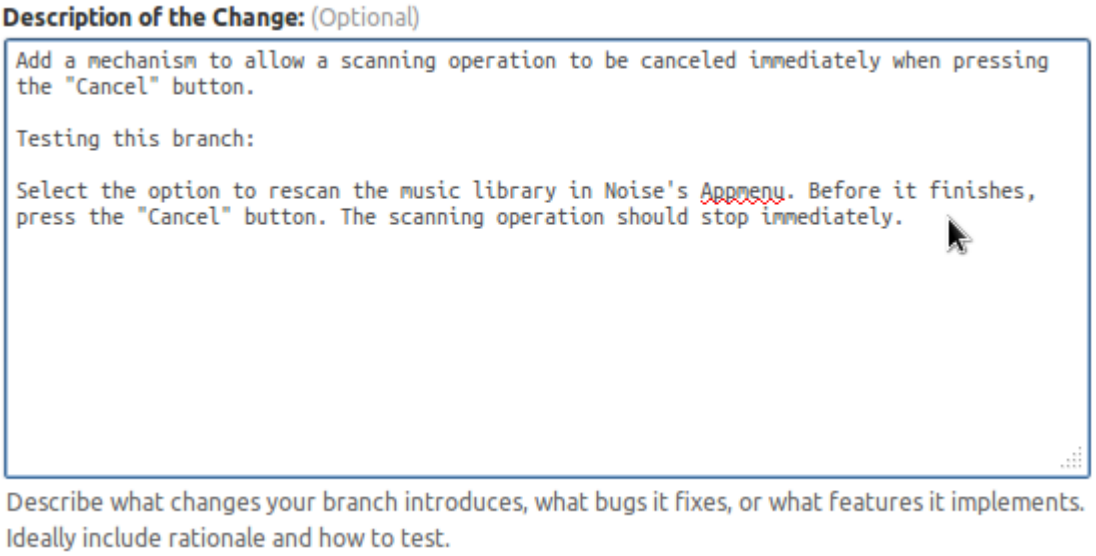
```
bzr lp-open
```



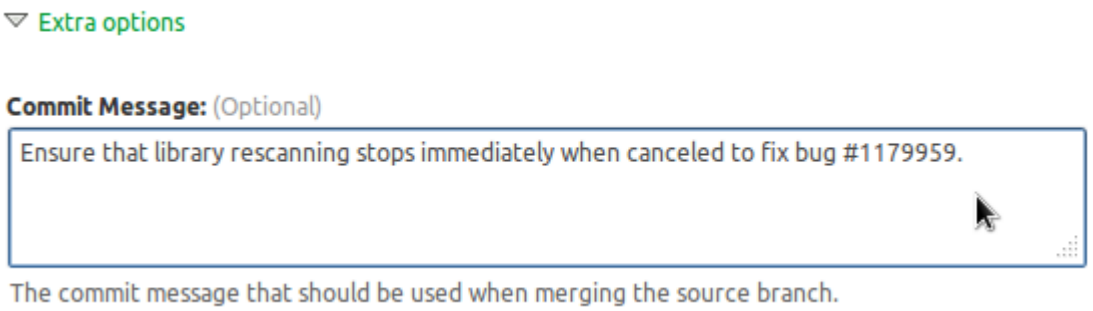
Click “Propose for merging”. You will need to complete the merge proposal with a couple of details before proceeding.



In “Description of the change” describe what changes your branch introduces, what bugs it fixes, or what features it implements. Ideally include rationale and how to test.



In “Commit message” (under “Extra options”) fill in with a concise summary of what your branch does. Typically, this will be the same as the commit message you used earlier in this guide.



After you completed your merge proposal, click “Propose merge”.

All that’s left for you is to wait for an elementary developer to review your proposed branch.

The reviewer may ask you to correct a defect in your code, or to give further explanations about what you did. All of these things will be notified to the email account that is tied to your Launchpad account. Be sure to keep an eye on your inbox!

After your branch gets approved, it will get merged automatically by a bot, but only if a commit message for the merge request was specified, so be sure to fill in the appropriate field.

#Tips for bug fixing

- If you have any doubts about the bug fixing procedure, or you need help for fixing a bug,

you can ask around in the elementary Development IRC channel (#elementary-dev at irc.freenode.net). To enter this channel, you can use our online client at <http://elementaryos.org/developer/chat>.

- The source code of all elementary projects follow the [elementary Code Style guidelines](http://elementaryos.org/docs/code/code-style), so be sure to check them out and to follow them as closely as possible.

- It's always a good idea to read whatever documentation came included with the project you are working on, for example README and HACKING files. These will usually contain instructions on how to properly test your program.

- Ensure that you are using the appropriate versions of the project's dependencies, for example selecting the appropriate "valac" version before compiling (valac 0.16.1 for Luna projects):

    sudo update-alternatives --config valac

- It may be a good idea to use the "~elementary-dev-community" user handle for pushing new branches. In this way, other people will be able to fix code style or minor errors of your proposal, reducing the time it takes to be accepted.

    push lp:~elementary-dev-community/projectname/fix-XXXXXX

Keep in mind you need to join the [~elementary-dev-community](https://launchpad.net/~elementary-dev-community) team to be able to do this.

- If someone fixed a bug and you want to contribute to that fix without taking credit, you can download that branch and fix the source, then upload your changes with the name and mail of the original developer:

    bzr commit --author="Bob Square <bobsquare@mail.com>" --fixes=lp:XXXXXX