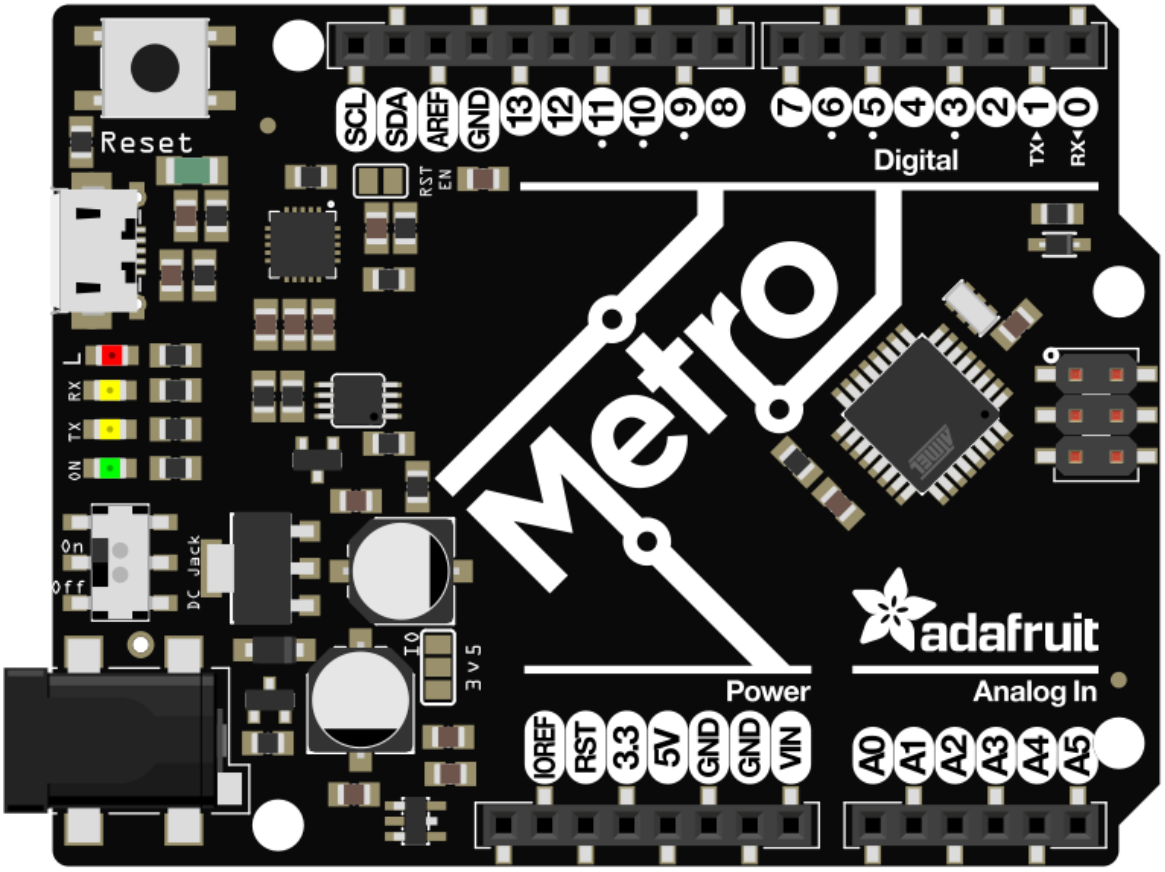
Make Beautiful Fritzing Parts with eagle2fritzing

Created by lady ada



https://learn.adafruit.com/make-beautiful-fritzing-parts-with-eagle2fritzing-brd2svg

Last updated on 2023-08-29 03:11:04 PM EDT

Table of Contents

Overview 3

Get Ready: Download Fritzing 7

Get Ready: Git Clone 7

- Step 1 Parent folder

Get Ready: Fonts 9

Get Ready: Inkscape 11

Get Ready: Qt Creator IDE 12

Get Ready: run.py edits 16

Run! 17

Check and customize PCB View 19

Check and customize Schematic View 24

Check and Customize Breadboard View 26

- Missing Subparts
- Create a new SubPart with Inkscape
- Moving Subparts
- PCB Color

Edit and Fix Fritzing File 34

- Breadboard Pin Gender
- Removing Extra Connections
- PCB View Review
- Schematic View Review
- Metadata

Terminal Blocks 40

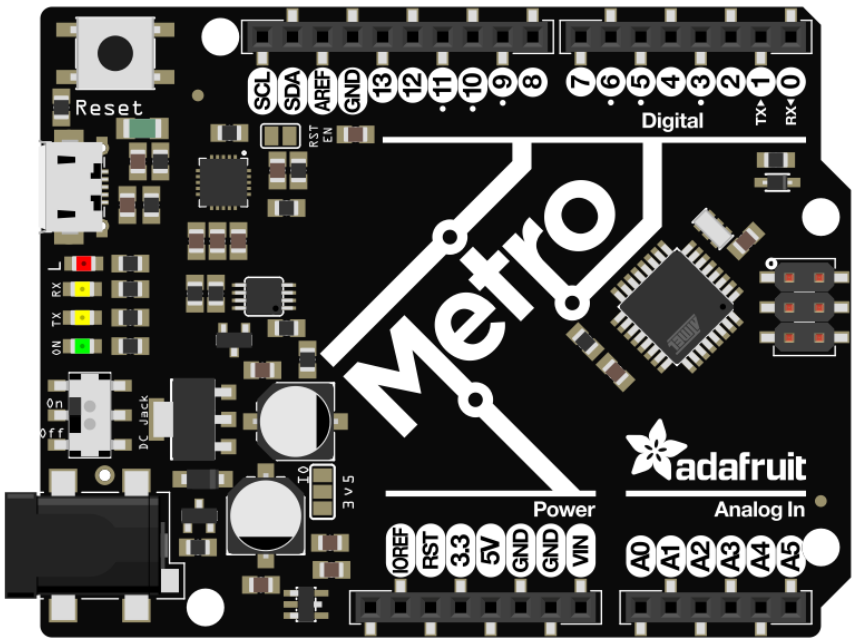
Test, Export, and Re-Edit Part 41

- Export Fritzing Part
- Editing Post-Creation

Best Practices! 46

- Rename file before conversion!
- Remove pin numbers
- Rename Signals in Eagle
- Rearrange pins in schematic

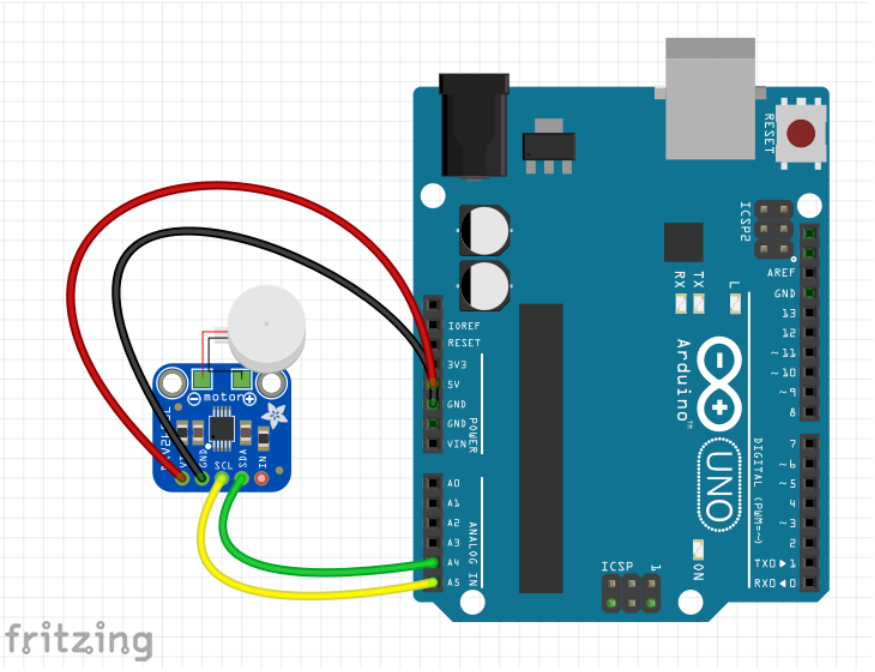
Overview



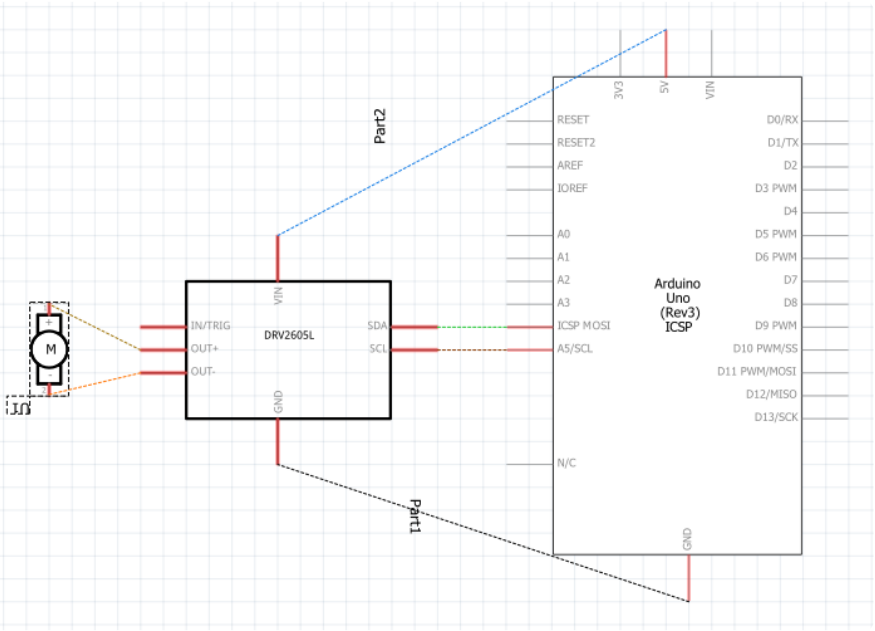
If you design maker hardware - you'll find investing in Fritzing objects to be an excellent method for encouraging projects. It'll also make your documentation look really nice.

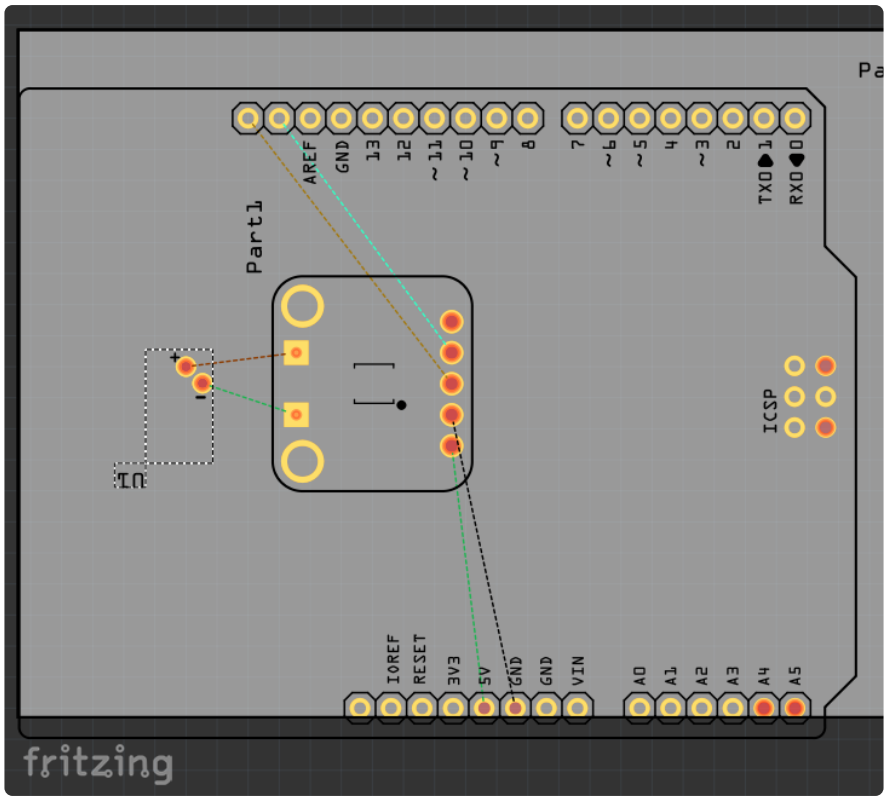
Fritzing is an open source CAD-like graphical software for all operating systems that allows full interaction between breadboard, schematic and PCB views. Customers or other enthusiasts can wire up breadboard diagrams and then turn that into a schematic and even a finished PCB!

For example, here's a demo of the part we'll be making, wired to an Arduino and buzzer in breadboard view:



And the corresponding schematic & PCB views

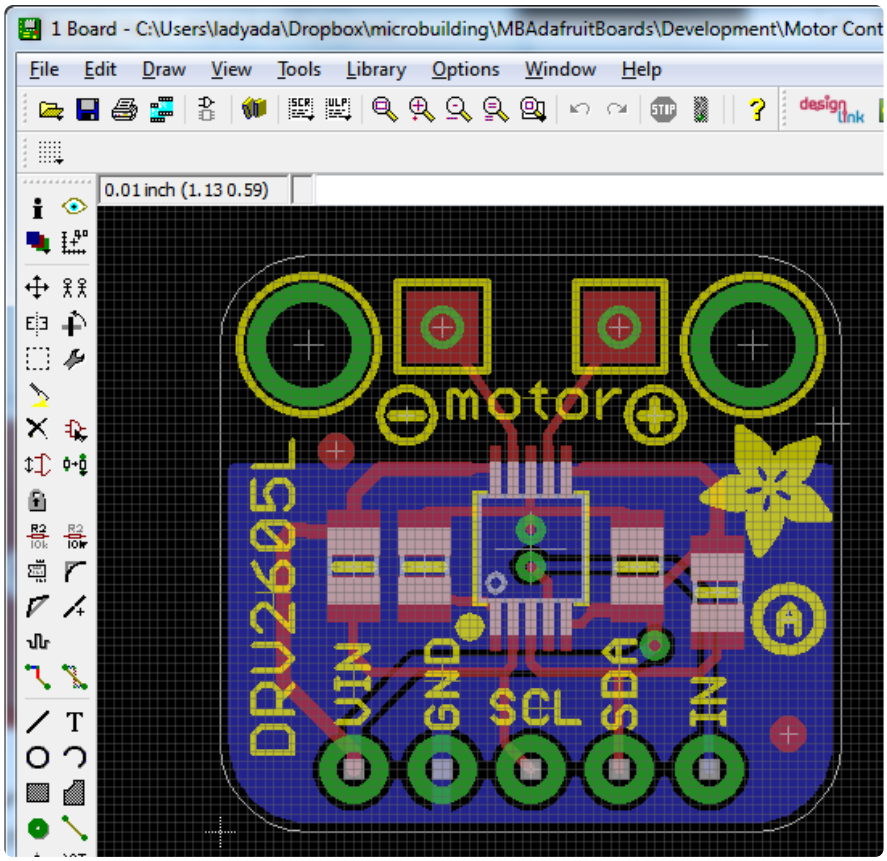




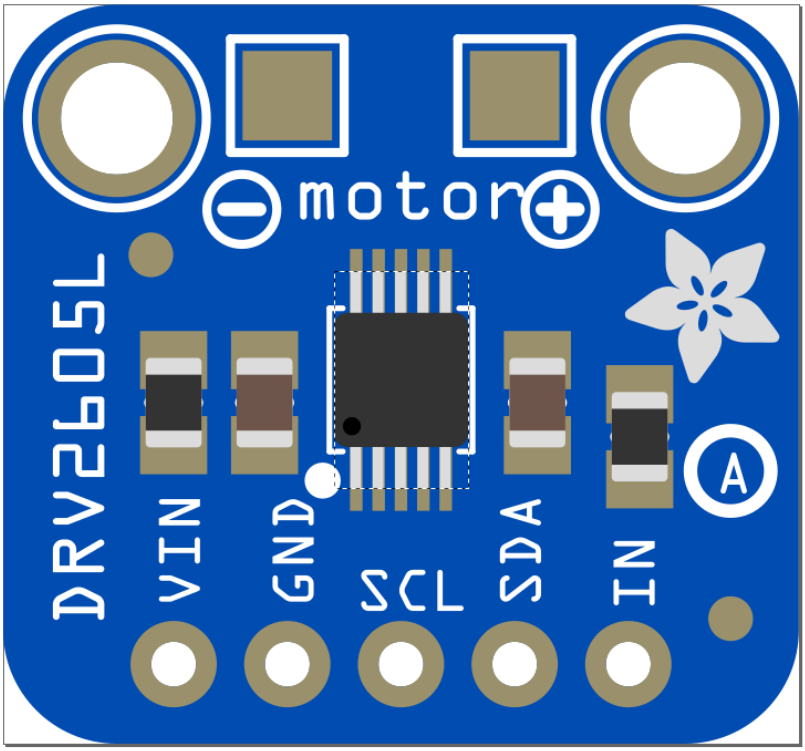
The idea is great but the making of those lovely objects is not easy. Here's some tutorials on how to make objects from scratch:

- Sparkfun's Make your own Fritzing Part ()
- Details on the Fritzing Parts Editor ()
- Fritzing's tutorials (some are out of date) lots of links there too! ()

These are all good for making custom parts but wouldn't it be really cool if we could take this:



A PCB file in EagleCAD format...and wave a magic wand to turn it into something like this?



A perfect 1:1 part, based on the CAD file?

WELL YOU CAN! (sorta)

Using the fancy eagle2fritzing project by Fritzing team and the excessive effort of PaintYourDragon () who wrassled with the code and got it into a really good spot!

This toolset works with any OS, in theory. Its been tested on Mac OSX and Windows 7 x64 and the tutorial is writted on Win7 so some variation may be required to get it working on your setup. Srsly.

HERE BE DRAGONS! eagle2fritzing can be very nerve-wracking at times. We did get it to generate our lovely parts but time and coaxing is required! It is not for people who are unable to put in the many (possibly frustrating) hours to learn the quirks and tweaks necessary for really advanced usage! We offer absolutely no support what-so-ever for this tutorial, you are on your own....You have been warned!

Get Ready: Download Fritzing

Of course you'll need to download the Fritzing desktop app in order to try out the parts!

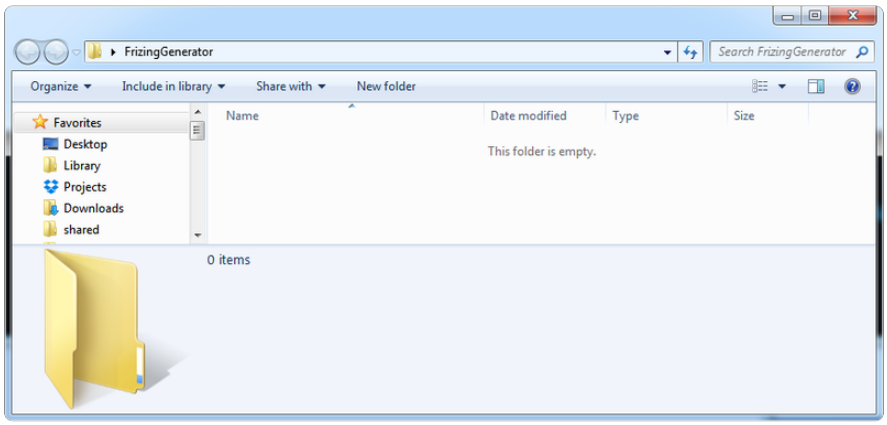
You can download the latest version from http://fritzing.org/download/ ()

Get Ready: Git Clone

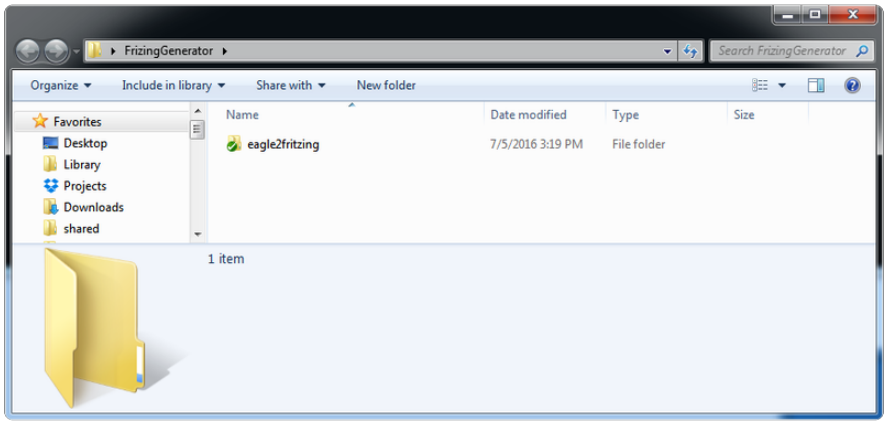
Your first step is to create the directory where you will work - its where both code and parts live. This assumes you are using git but you could in theory just download them. If you do use git, you can fork and submit pull requests for subparts and more. We don't cover git here, its assumed you know how to use it.

Step 1 Parent folder

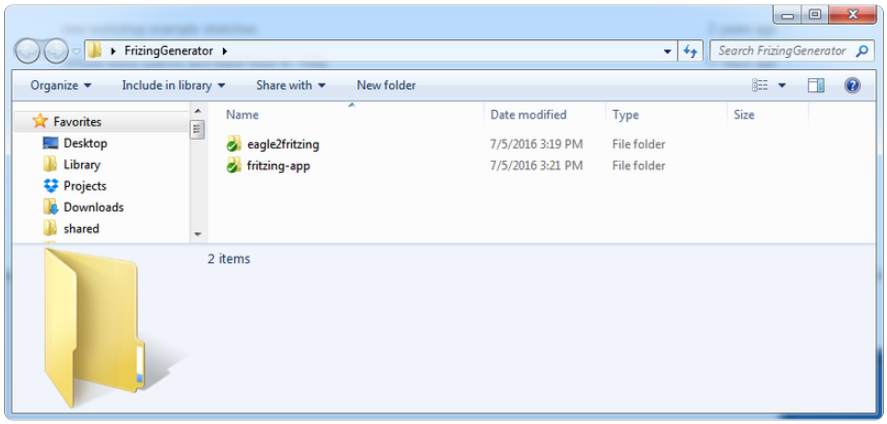
Create a new parent folder anywhere you like



Clone https://github.com/adafruit/eagle2fritzing () into that directory



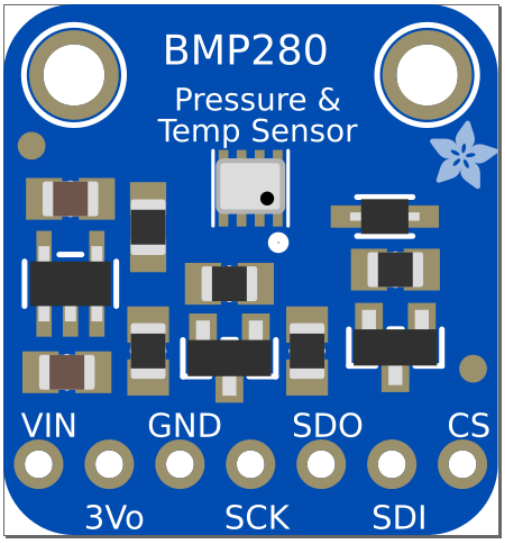
Clone https://github.com/fritzing/fritzing-app () into the same directory



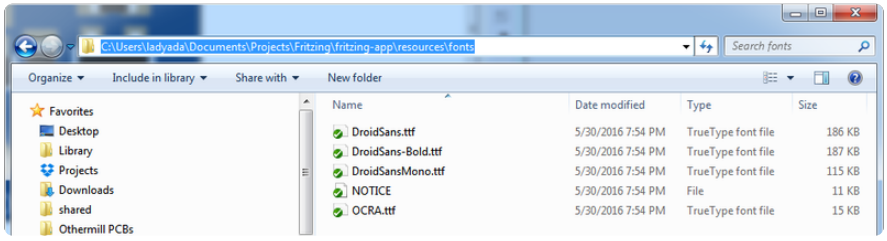
Get Ready: Fonts

A nice realism of the Fritzing converter is it takes all the silkscreen text and orients it in the right location for you and in the right size

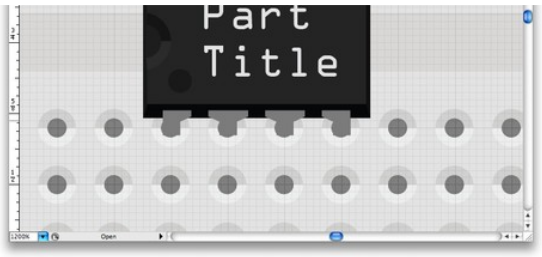
However, without the right font, your part will default to a rounded sans serif:



You can find the right fonts in the fritzing-app/resources/fonts folder



You can also download the zip with fonts from http://fritzing.org/fritzings-graphic-standards/download-fonts-and-templates ()



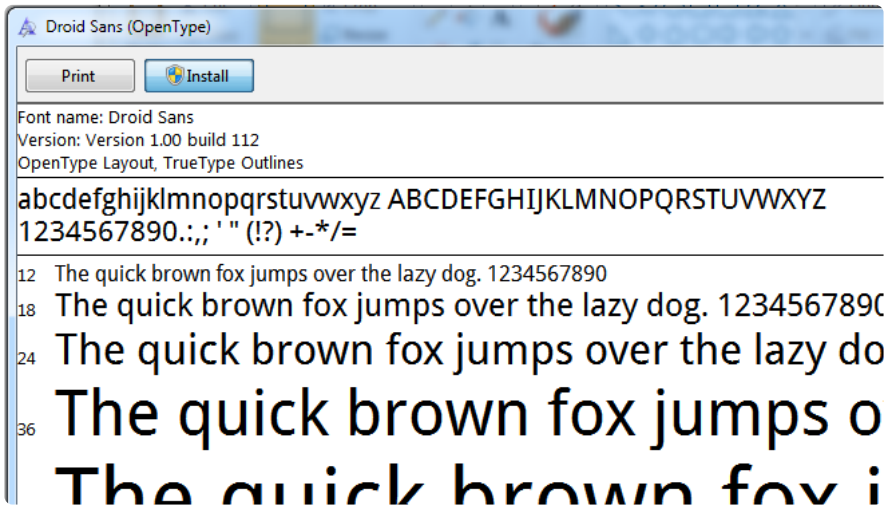
You can download and use these helpful templates to design your custom parts according to Fritzing's Graphic Standards.

The following .zip file includes:

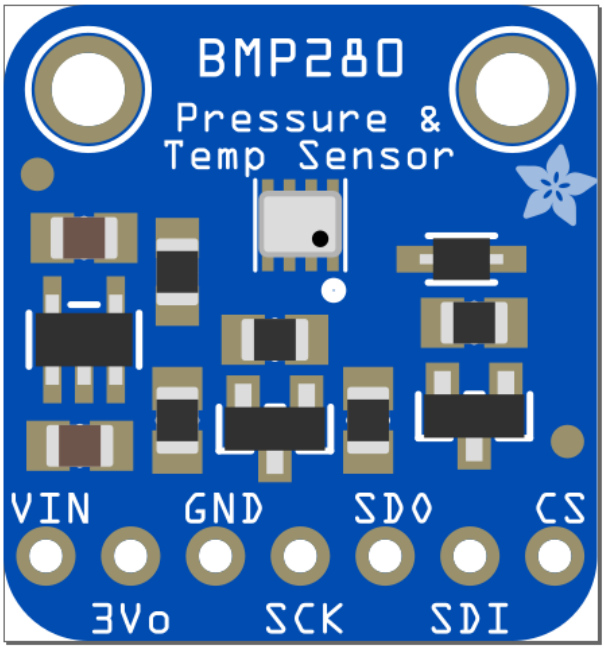
1. Droid Sans font
2. OCRA font
3. Breadboard View graphics template
4. Schematic View graphics template
5. PCB View graphics template

Download .zip file

Install all four TrueType (TTF) fonts!



You don't have to regenerate the svg, but you may need to restart inkscape



Get Ready: Inkscape

You will need to edit some SVGs and the best tool we've found for that is Inkscape. You could also use CorelDraw or Illustrator but Inkscape works really well and once you get the flow down is very quick.

You can download it from https://inkscape.org/ ()

It's available in binary format for all operating systems! Yay!

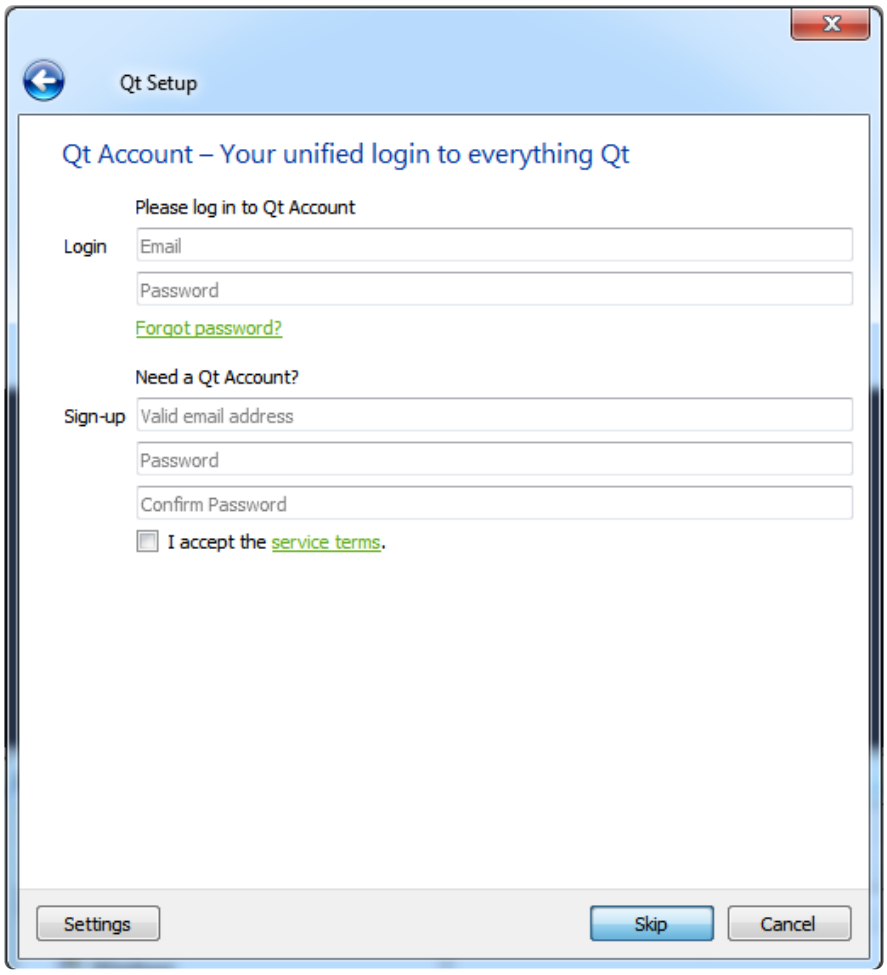
Get Ready: Qt Creator IDE

We have precompiled executables but if you're not using Windows 7 64 bit you may need to recompile the brd2svg program. This program is written in Qt!

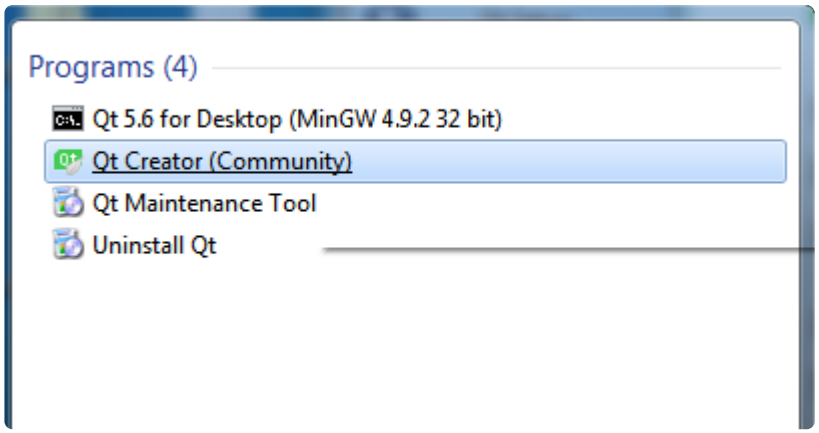
We told you this would be fun right?

You can grab the compiler/IDE from https://www.qt.io/download-open-source/ ()

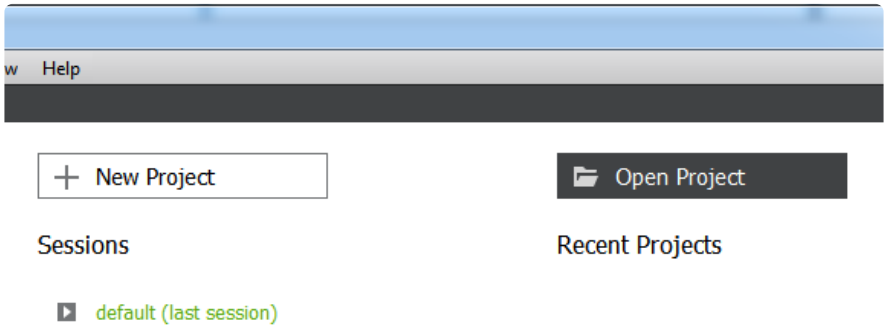
You can skip the Qt account setup part



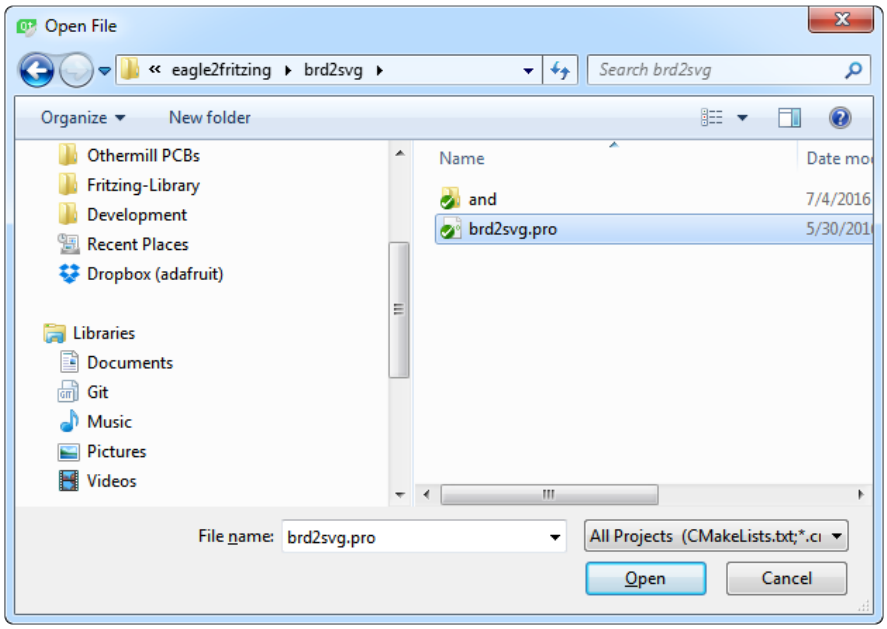
Once installed, run Qt Creator



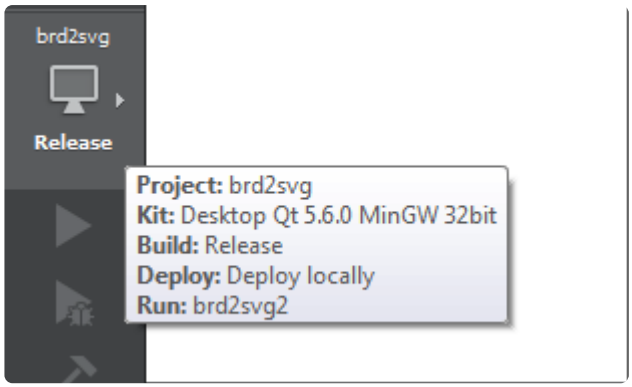
Open up the project:



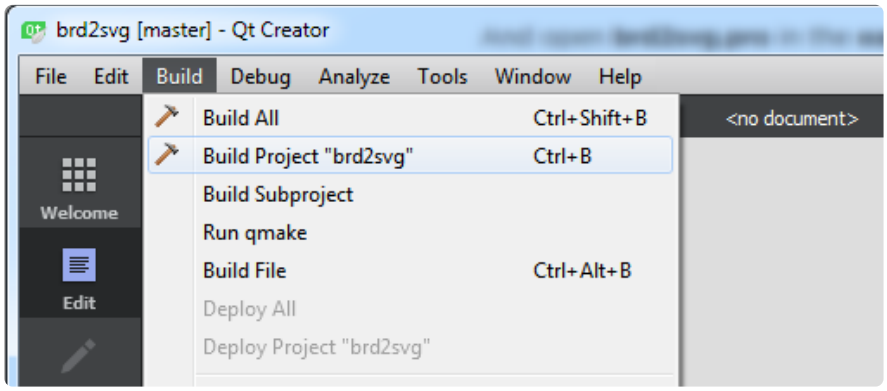
And open brd2svg.pro in the eagle2fritzing/brd2svg folder



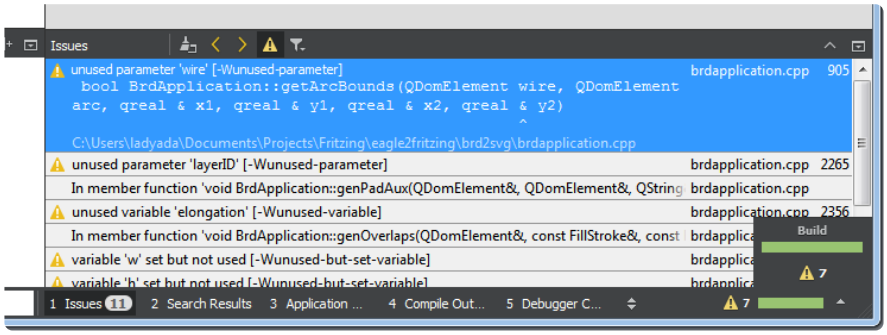
Select Release build mode



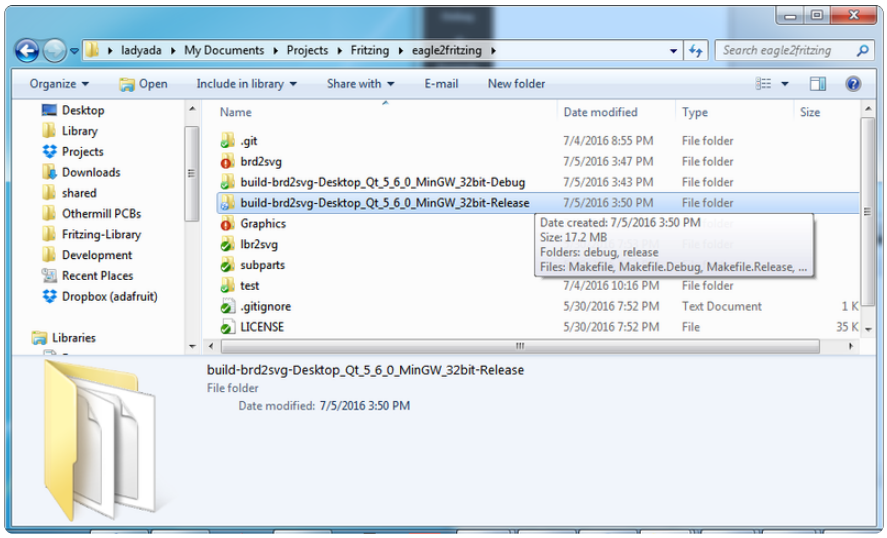
Build the project



You may get warnings, thats OK - the software is still 'in progess'

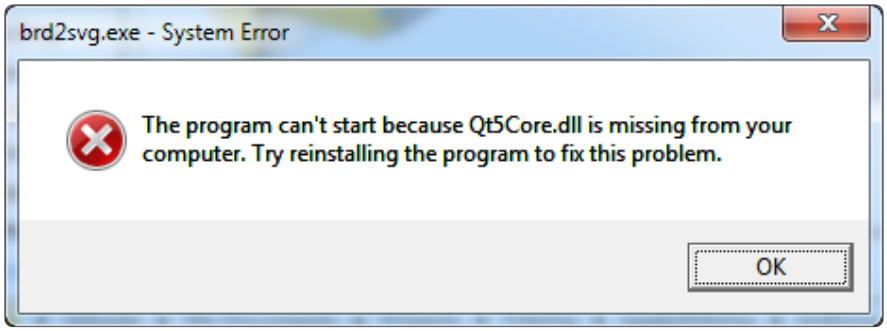


Once done you'll have a new build directory:



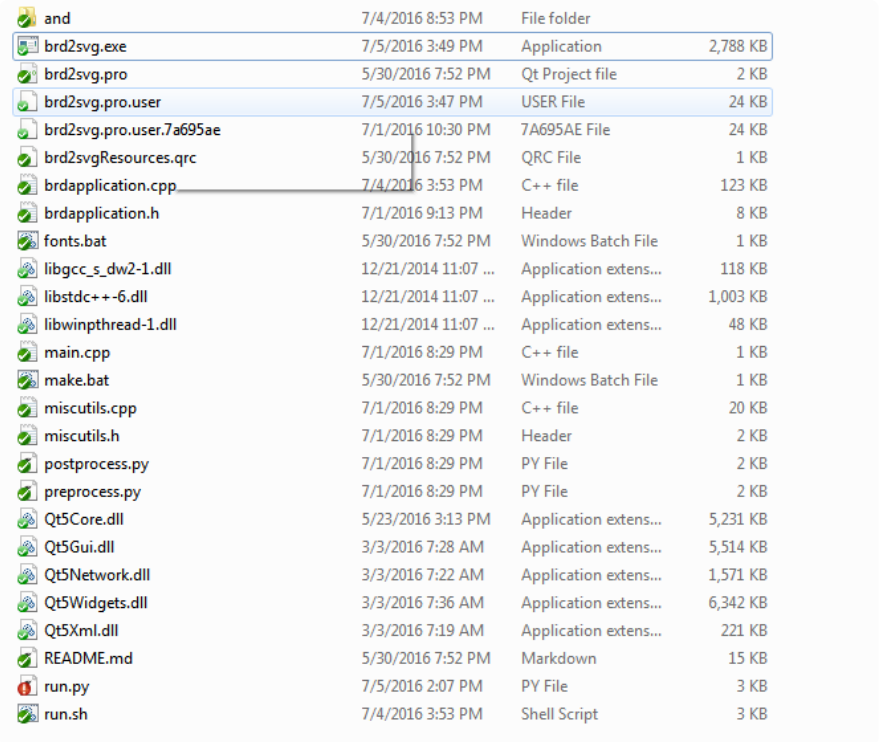
which inside the release directory you'll find your brd2svg executable. Drag this into the eagle2fritzing/brd2svg directory.

If you run it you may get complaints about missing DLL's

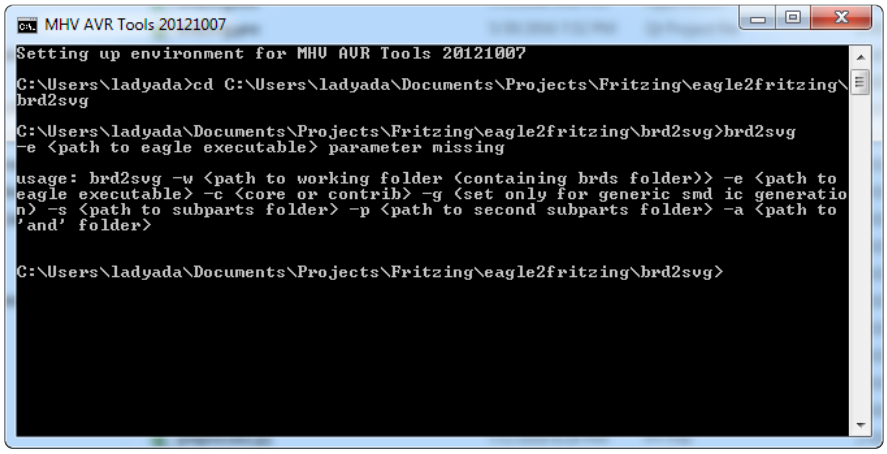


You can find these at C:\Qt\5.6\mingw49_32\bin

Just drag one at a time until you can run it without complaint



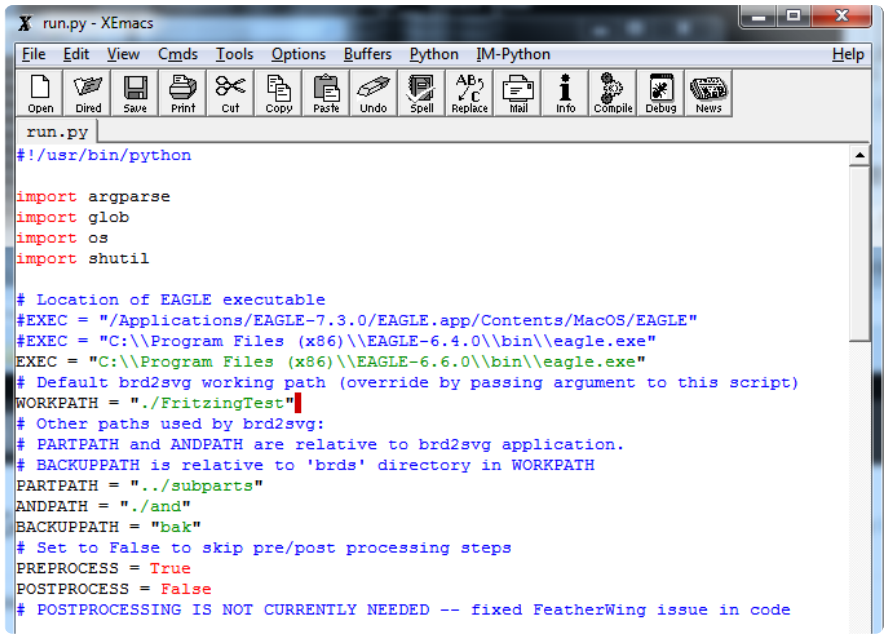
Then from within a terminal command line you can run the tool. It will complain it needs files. This is OK! If you get this warning, you can move to the next step



Get Ready: run.py edits

We have helpers on top of helpers to get the EagleCAD board file -> XML -> FPZ

The runner is a program that will set up your command line and do preprocessing for you. Edit run.py in the brd2svg directory:



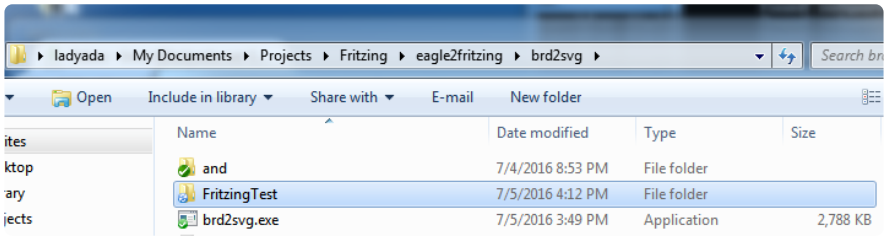
You'll definitely need to edit the EXECPATH to where eaglecad lives on your computer

You can also change WORKPATH later if you need

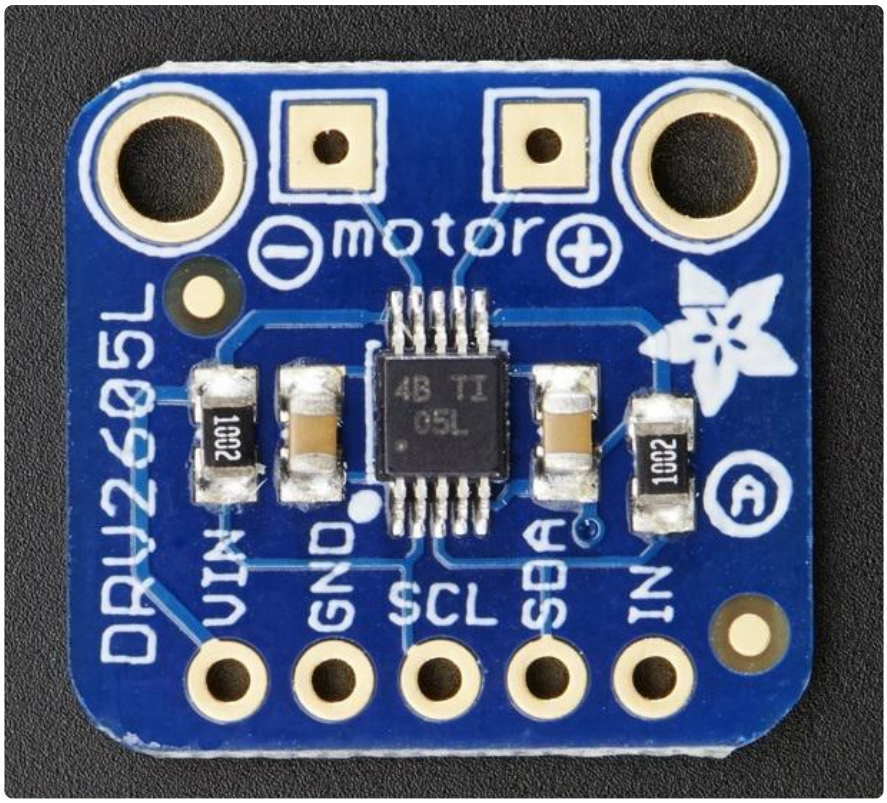
Now, finally, we can run this tool! ARE YOU READY?

Run!

Create your FritzingTest work folder in brd2svg and create a brds folder inside of that.



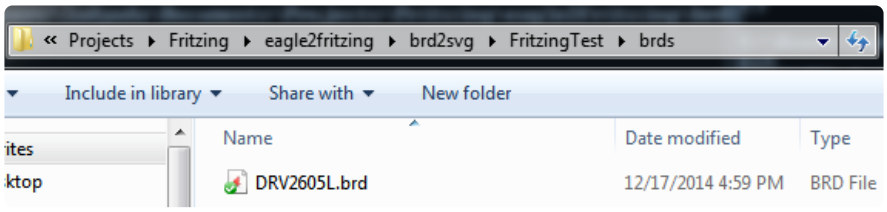
Add a brd file, in this case lets do the DRV2605 Breakout:



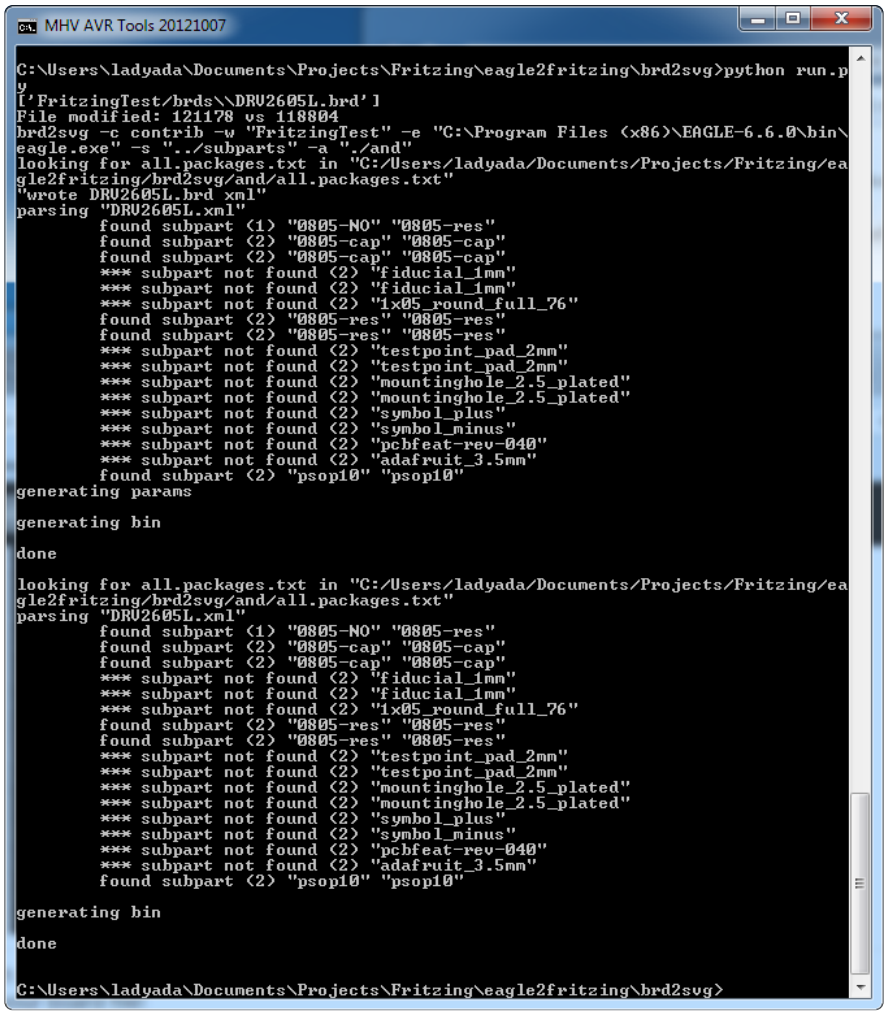
The PCB files are in GitHub here ()

You don't need to add the SCH file. Also, note that the board file will be modified so only work on a copy of your board file!

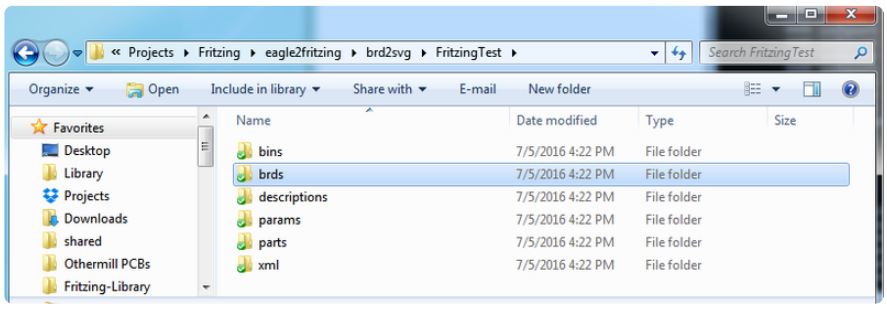
Don't have spaces or - or other than A-Z and numeric and _ in the file name, it will avoid parser-choking



Run python run.py in the brd2svg directory. It will run the preprocessor and application once or twice:



You'll now have all the part files in FritzingTest

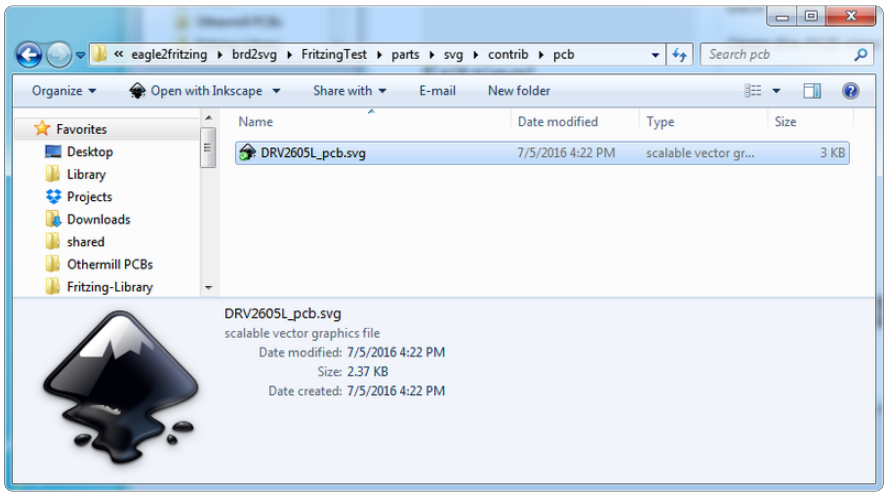


Check and customize PCB View

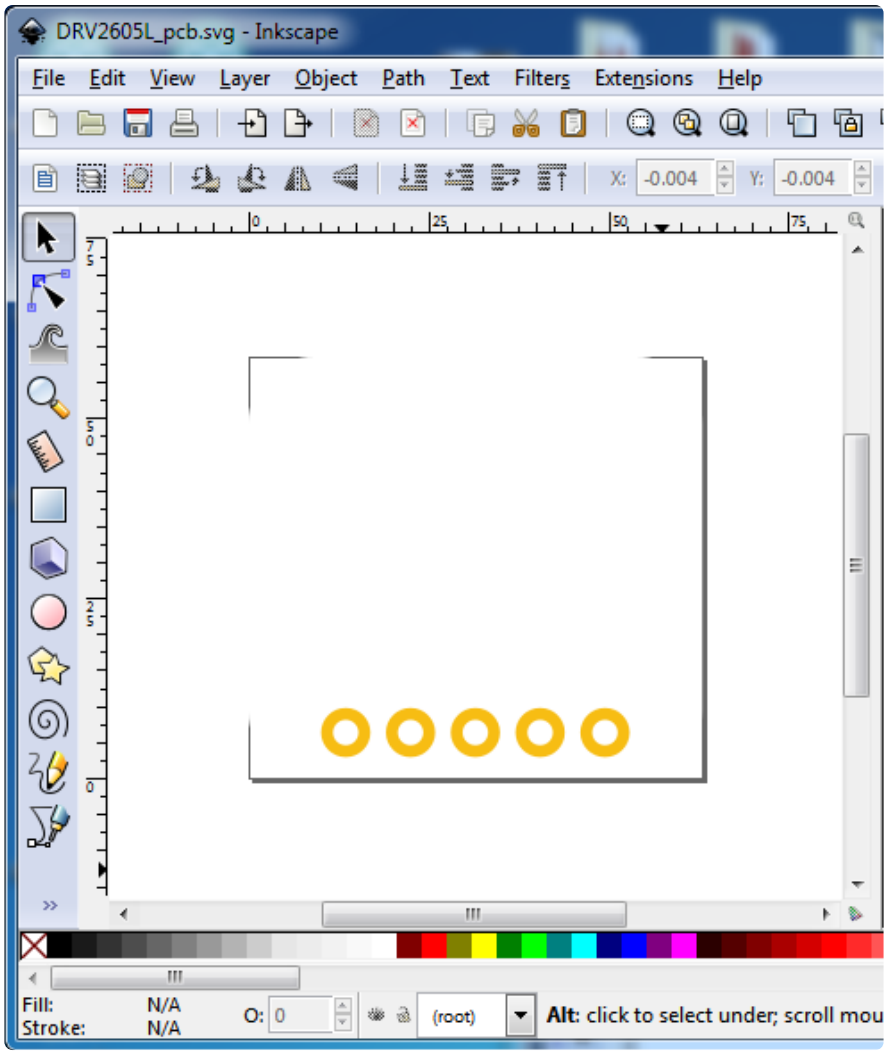
Fritzing PCB SVGs are very finicky about having the components in the right groups and right group names. Don't ungroup parts, instead try to edit/enter the group to keep the part and group names the same. Breadboard and Schematic are not picky, you can ungroup without issue

We suggest working 'backwards' - getting PCB view to be exactly the way you like, then working back to the breadboard view.

Open the PCB view SVG in the FritzingTest/parts/svg/contrib/pcb

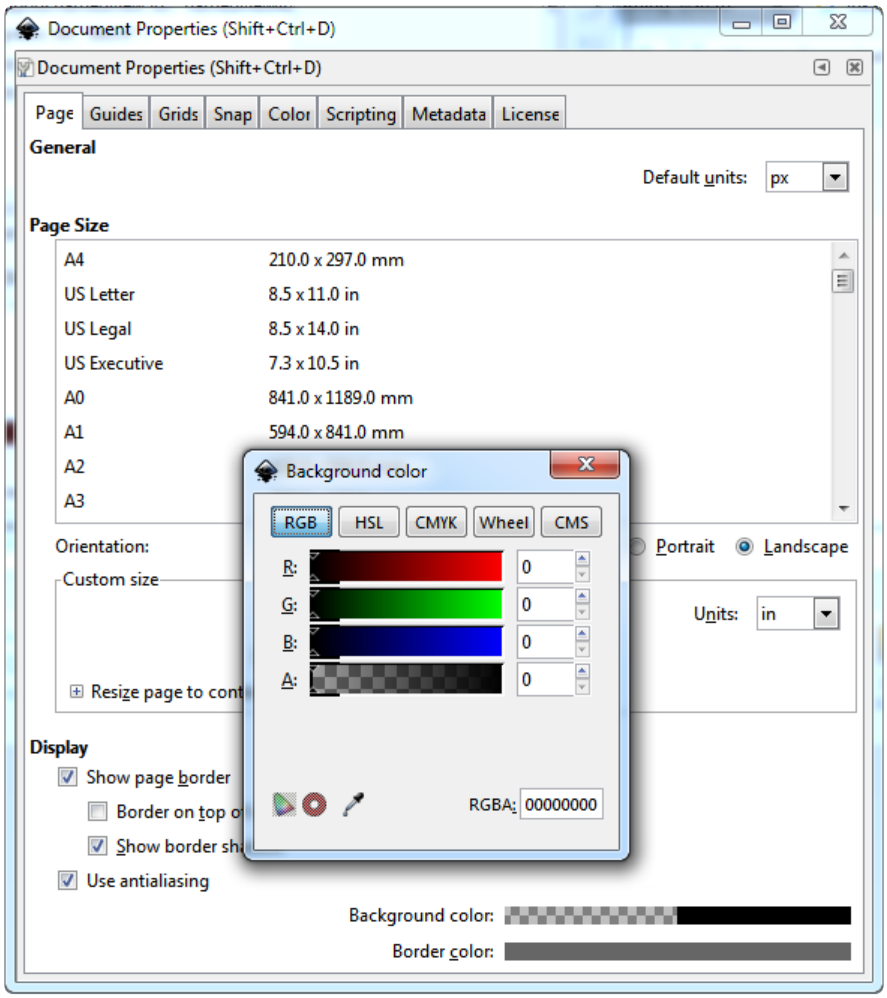


It'll look a little like:

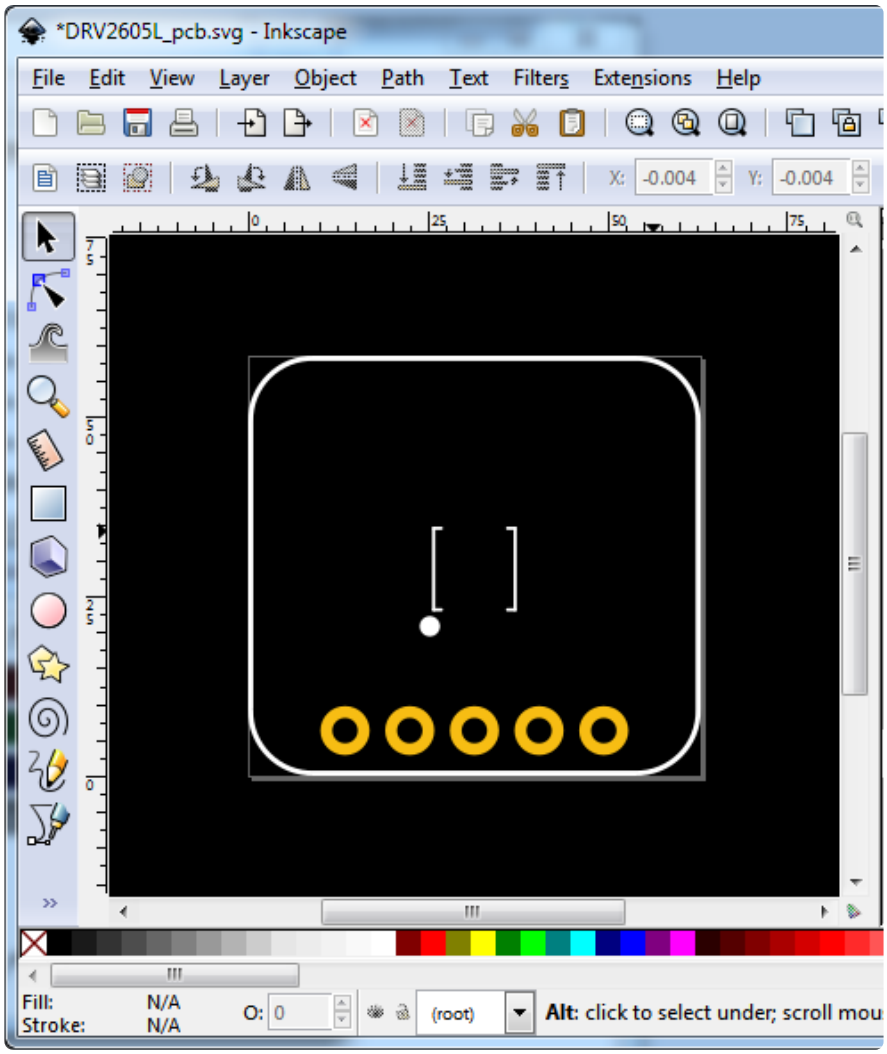


Since the lines are in white, you will want to change the background color. Open up the Document details with Control-Shift-D

Select Background Color and change it to black

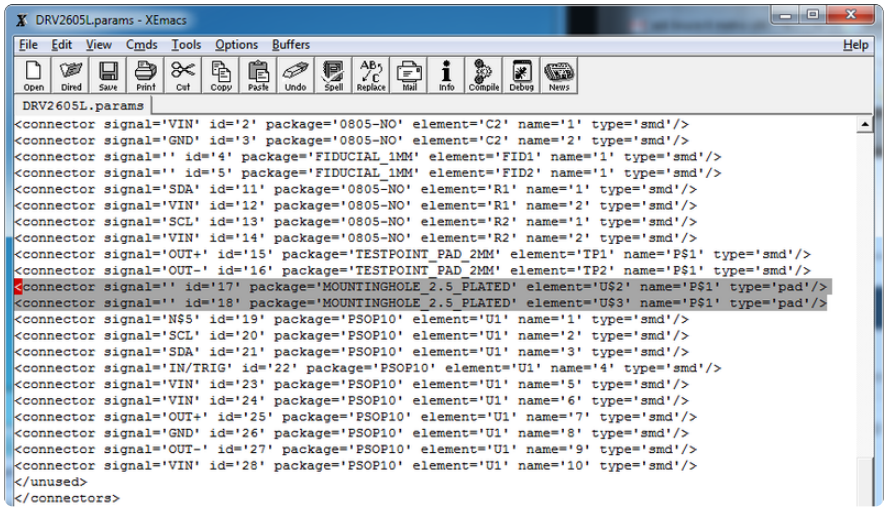


To see all the outlines:

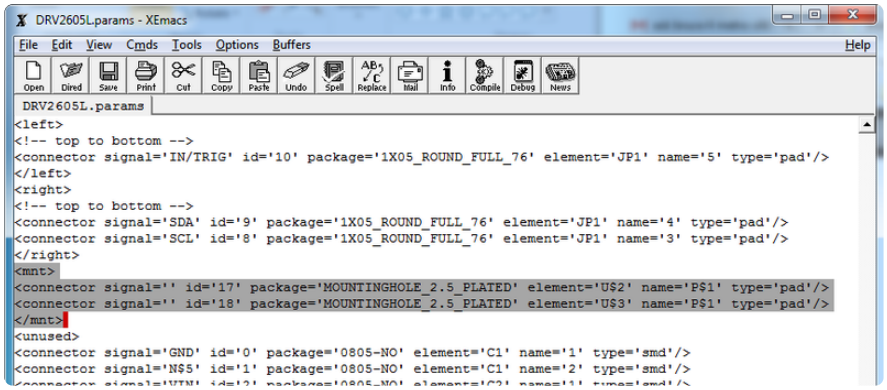


I would like the mounting holes to appear. To do that, you'll need to edit the part parameters and re-run the tool.

Edit FritzingTest/params/part.params and find the connector that ties to the mounting holes. The holes aren't actually connectors but this is the easiest way to do it

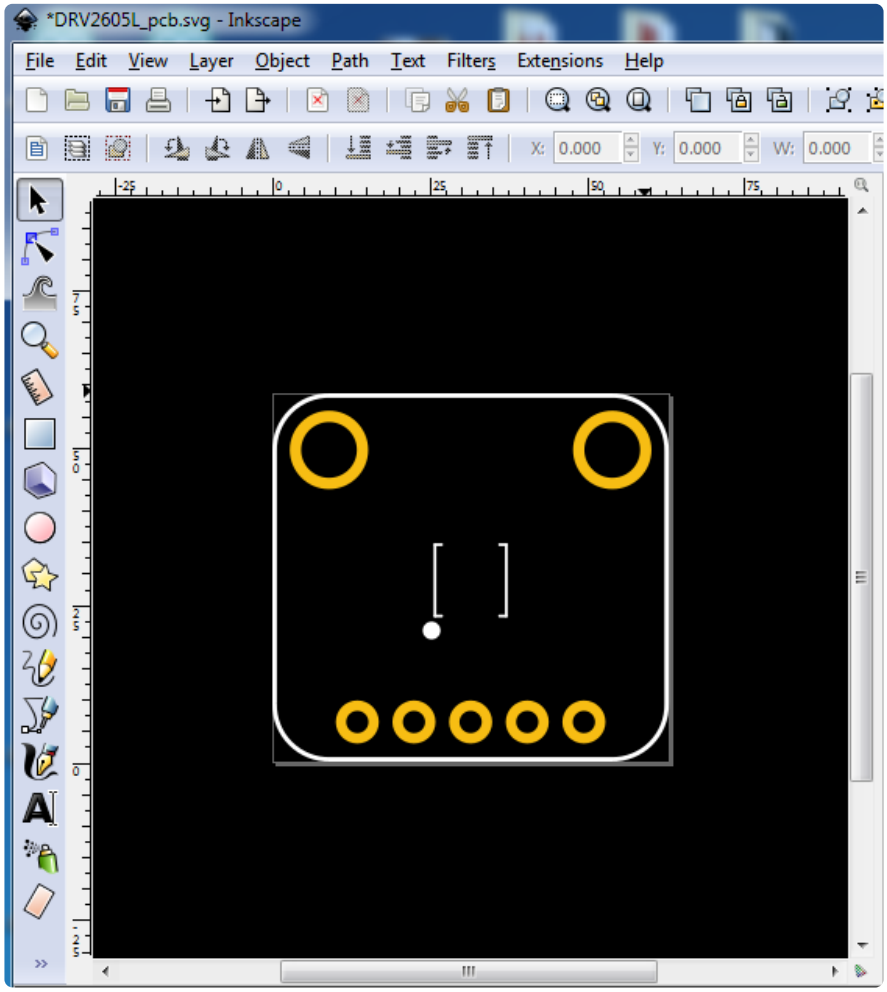


Move those out from the `<unused>` XML element into a new element you have to create called something like `<mnt>` (it can be anything really)

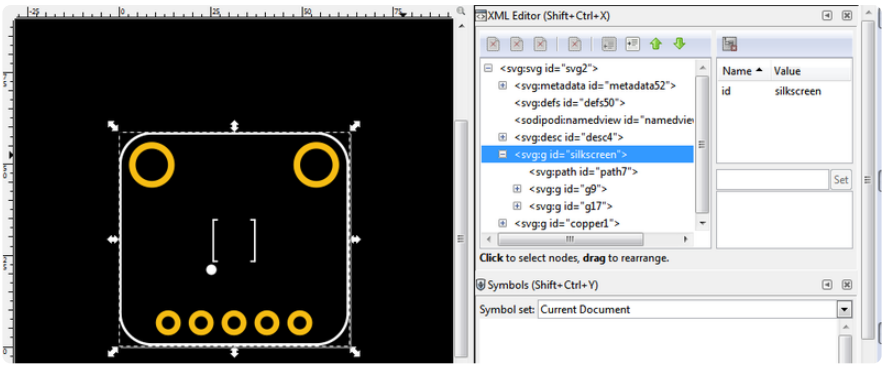


Save the params file and rerun the tool. It's also good to make a backup in case!

Mounting holes will appear



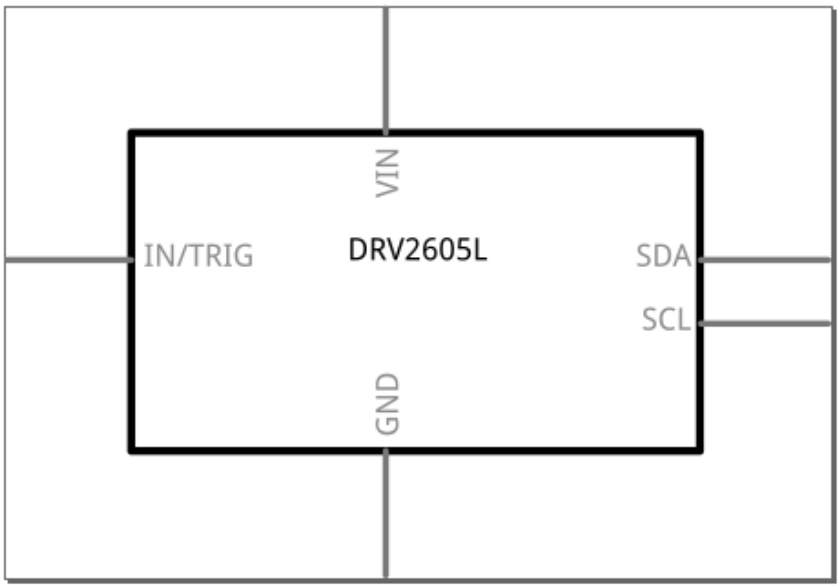
Editing the silkscreen is a little more advanced and isn't covered here. Basically make sure that all the silkscreen is in a group with id=silkscreen It's best to use the XML editor, use Shift-Ctrl-X to bring it up



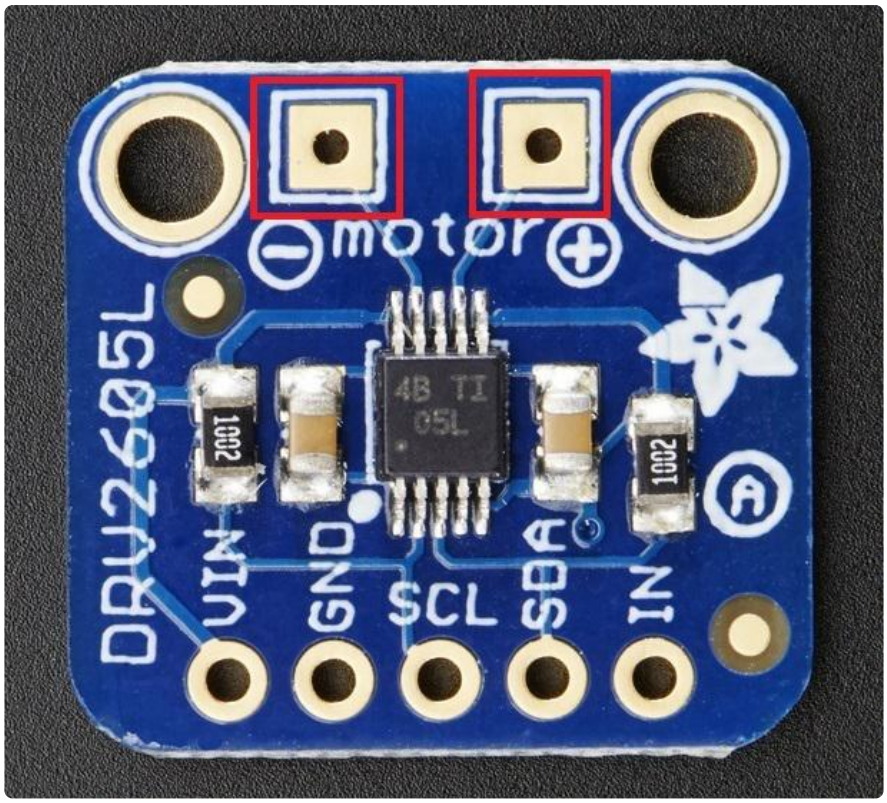
Once you have your PCB SVG in a good spot its a great idea to make a backup. Rerunning the brd2svg tool will OVERWRITE the file! Ideally you would not have to do any edits to the PCB file once params is set

Check and customize Schematic View

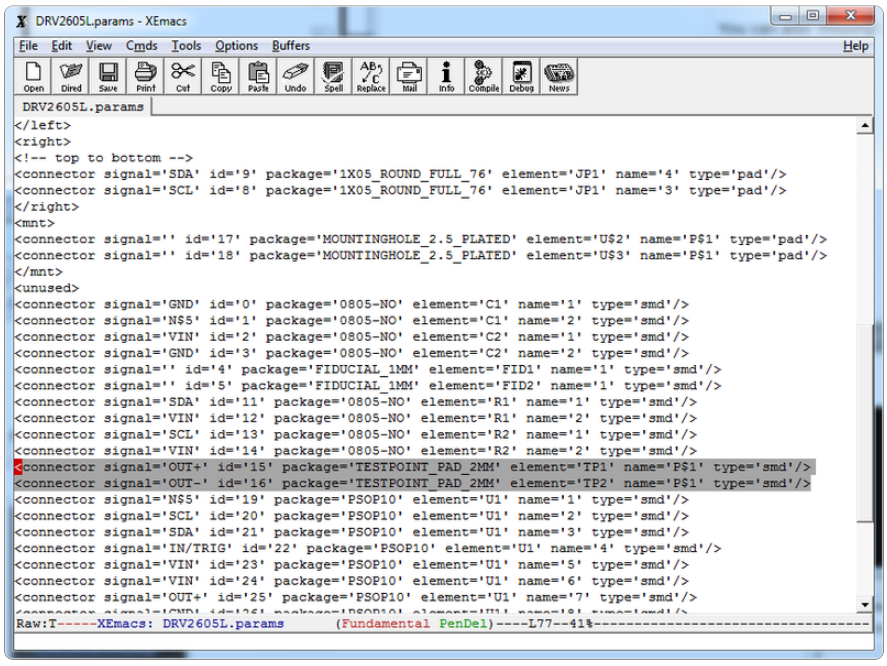
Once PCB view is in a good spot you can check the schematic svg. It is in FritzingTest\parts\svg\contrib\schematic



You'll notice that the svg converter is smart enough to make pins for all the through hole pads but did not create pins for the SMT motor pins



You can add 'missing' connectors easily. Go to params file again to find the two pads that are used for the motor:

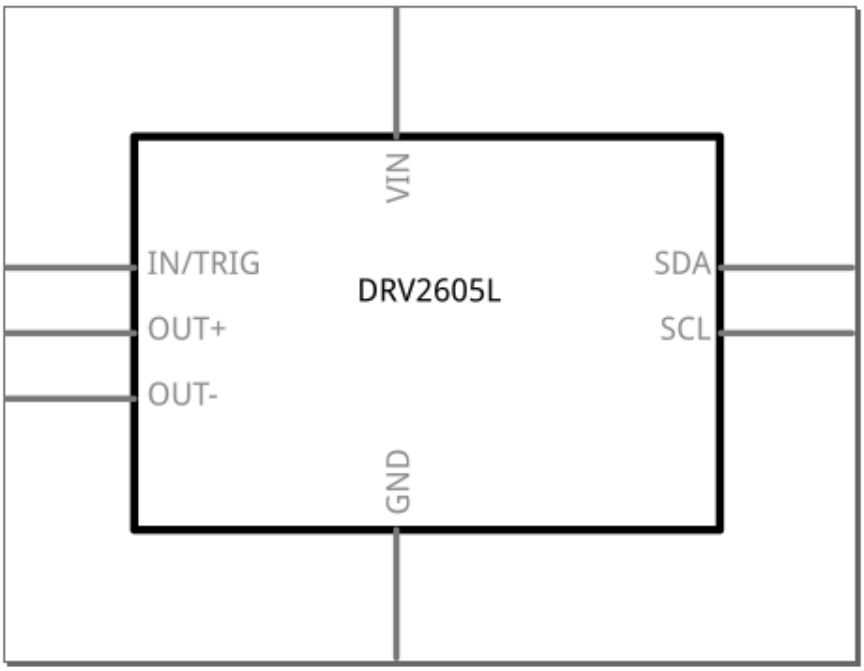


Cut these lines and place them in the group above, in the <left> xml group, that will put the pins on the left side.

Now is also a good time to rearrange the pin orders if you want them to appear in the schematic in a certain location or order

If you have multiple connectors with the same signal (e.g. GND) then the one with the LOWEST id='nn' number is the one that will appear

Rerun the tool to regen the schematic



You can hand edit the schematic and move around pins, but you'll need to reassign the pins later, just keep eveything on a 0.1" boundary to keep with convention. Since Fritzing isn't known as a hardcore schematic capture program we don't spend a ton of time on it.

You can edit the name/labels of the pins easily. Just make a backup because you'll rerun the tool to get breadboard view right, which will overwrite your changes! We find it a lot easier to update the signal names directly in EagleCAD, that way you aren't editing signal names in multiple SVGs

Once you have your Schematic SVG in a good spot its a great idea to make a backup. Rerunning the brd2svg tool will OVERWRITE the file! Ideally you would not have to do any edits to the Schematic file once params is set

Check and Customize Breadboard View

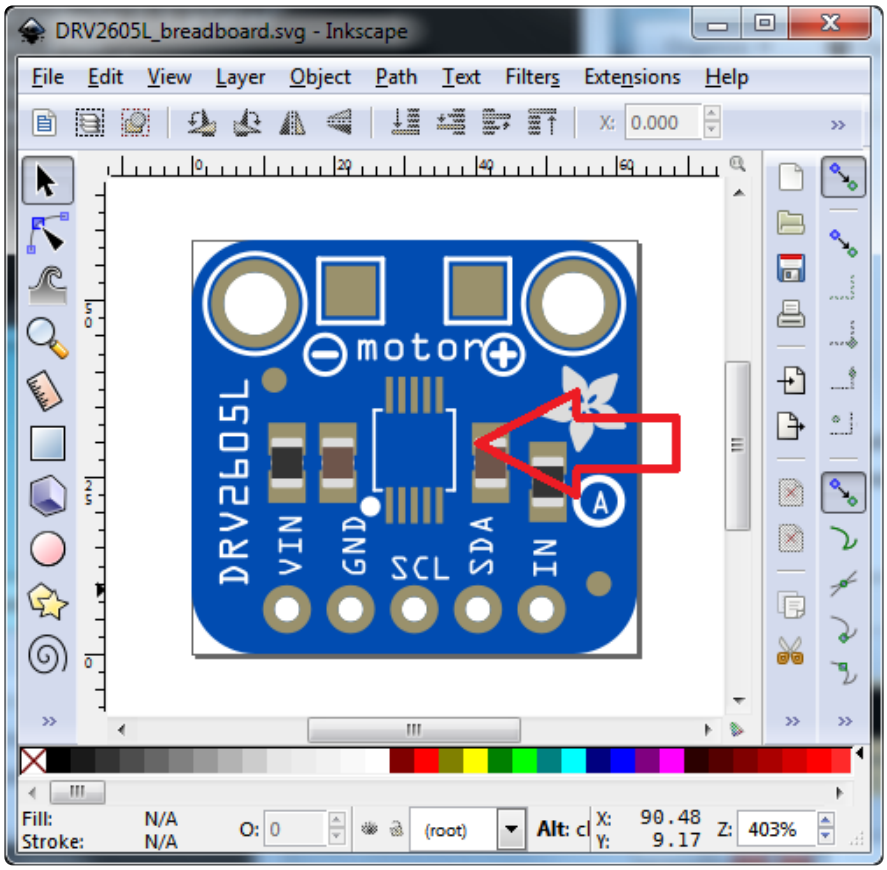
Now you are ready for the real fun - creating the breadboard view!

Open up the SVG in FritzingTest\parts\svg\contrib\breadboard

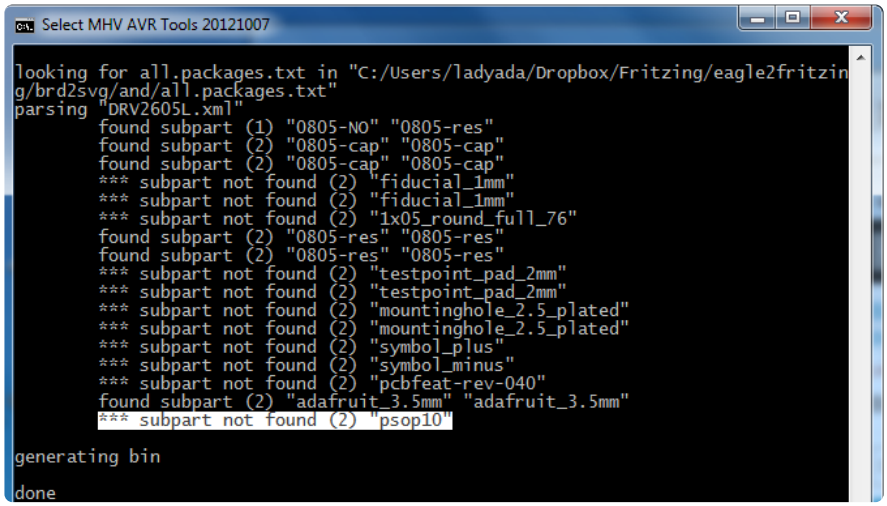
There's a large number of Things That Can Go Wrong. We'll cover as much as we can but you may have to do some of your own experimentation. You'll almost certainly need to do cleanup, but it wont be too bad!

Missing Subparts

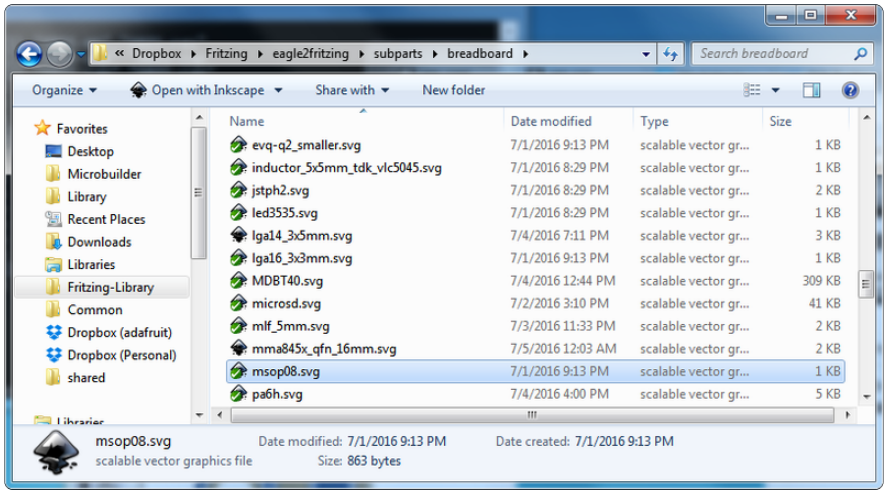
First thing you'll possibly deal with is missing sub-parts:



You can look at the output of brd2svg to see which parts were found and which were not



In this case, psop10 is the package that was not found. All the sub-components for your design are stored in eagle2fritzing\subparts



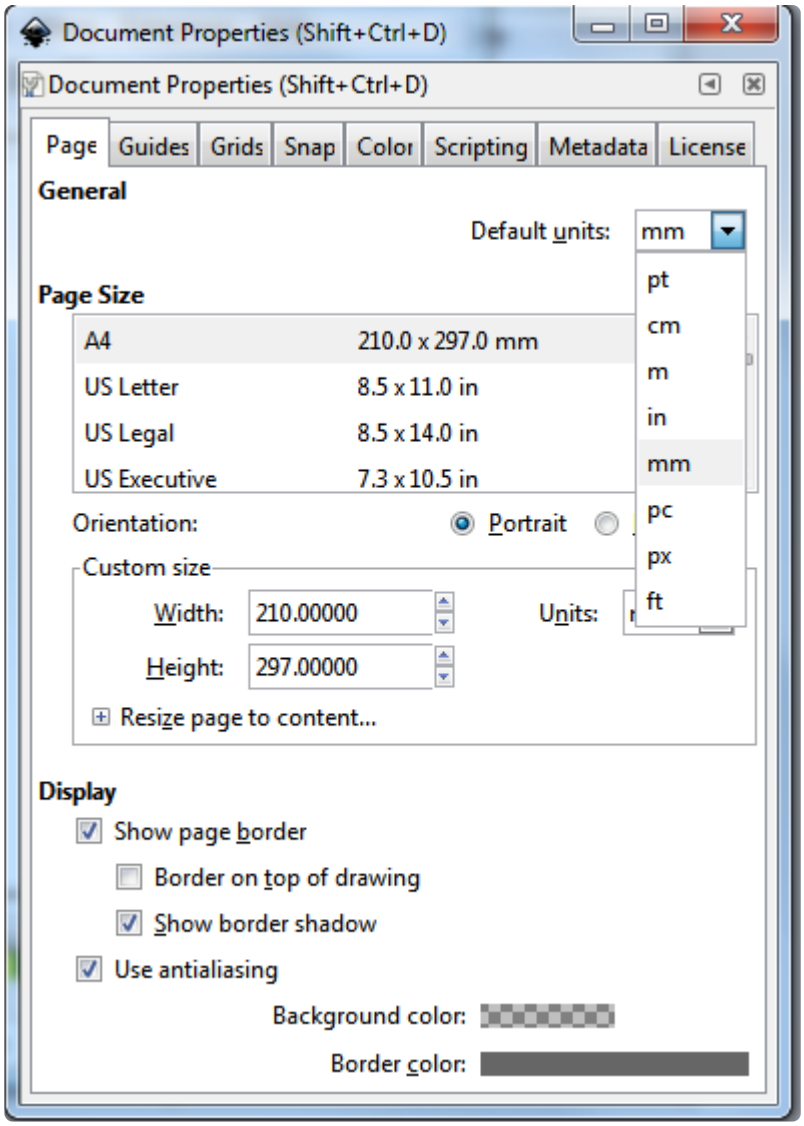
Not surprisingly, they are also all SVG! You can see in this example that we have an msop08 which is similar to a psop10 but not quite.

Lets create a new part. You can adjust an old part or start from scratch in Inkscape

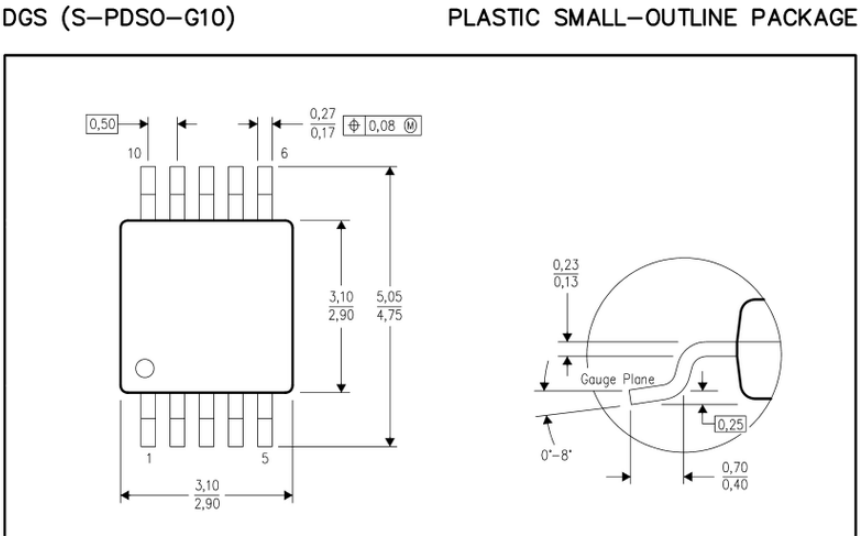
You can sometimes recycle subparts from other drawn Fritzing objects, check your Fritzing/parts/svg/core/breadboard (in the install folder) for core items that may have SVGs you can split apart for chips, capacitors, connectors and more!

Create a new SubPart with Inkscape

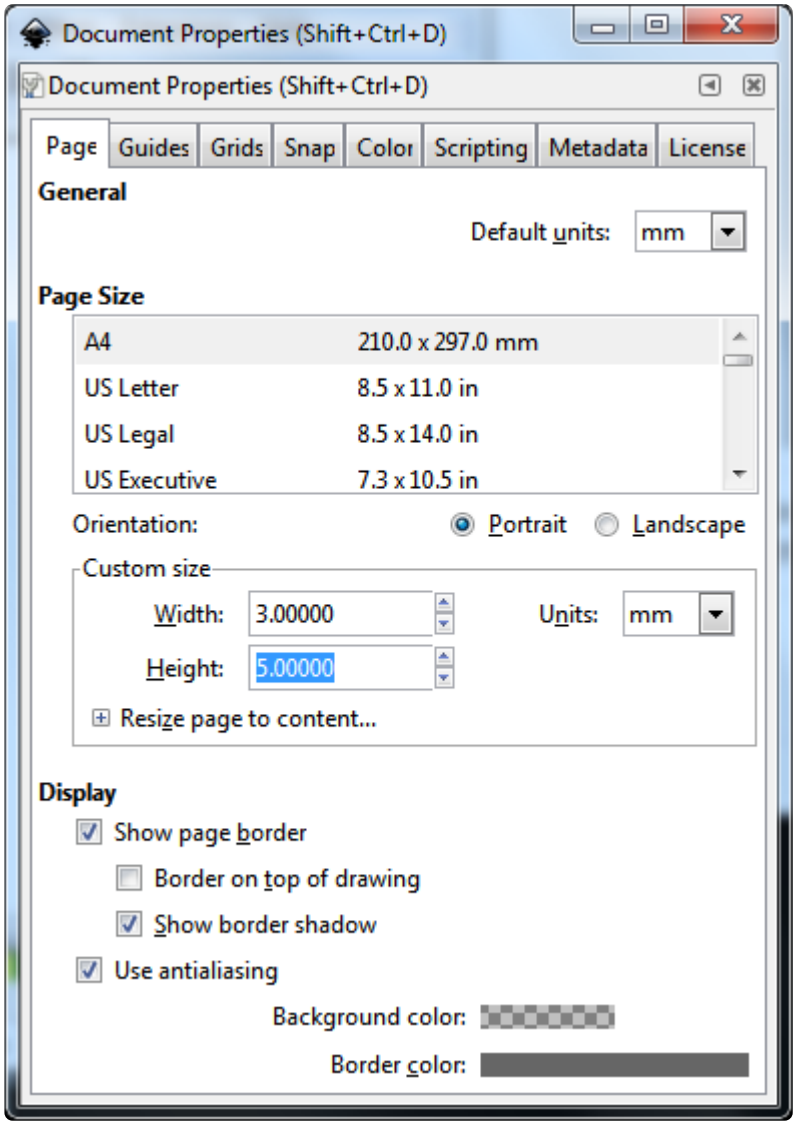
Start by opening up the Document settings (Ctrl-Shift-D) and setting the default units to mm (or inches on the off chance you have an inches-based part)



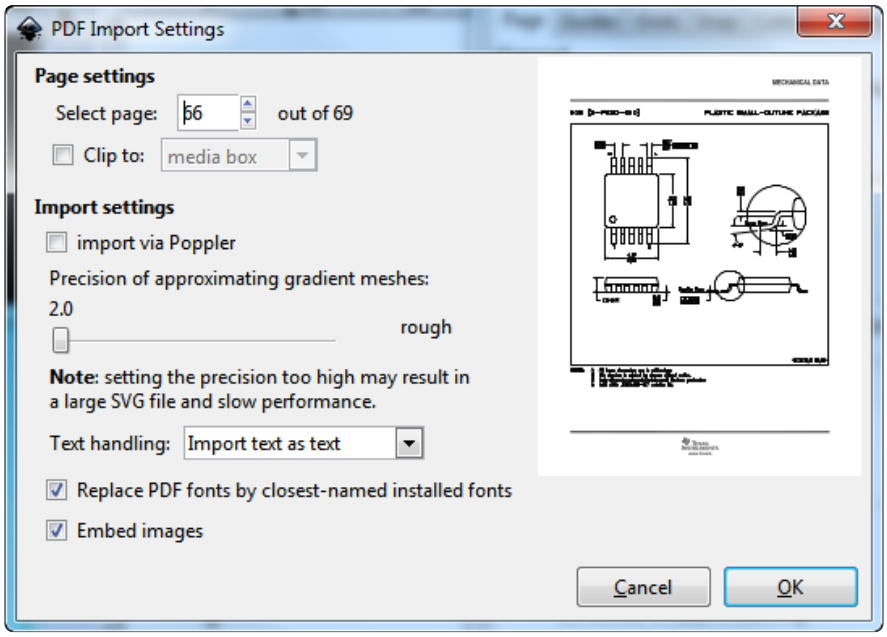
Check the datasheet for a diagram, which is handy even if you can't directly use it since it has all the measurements.



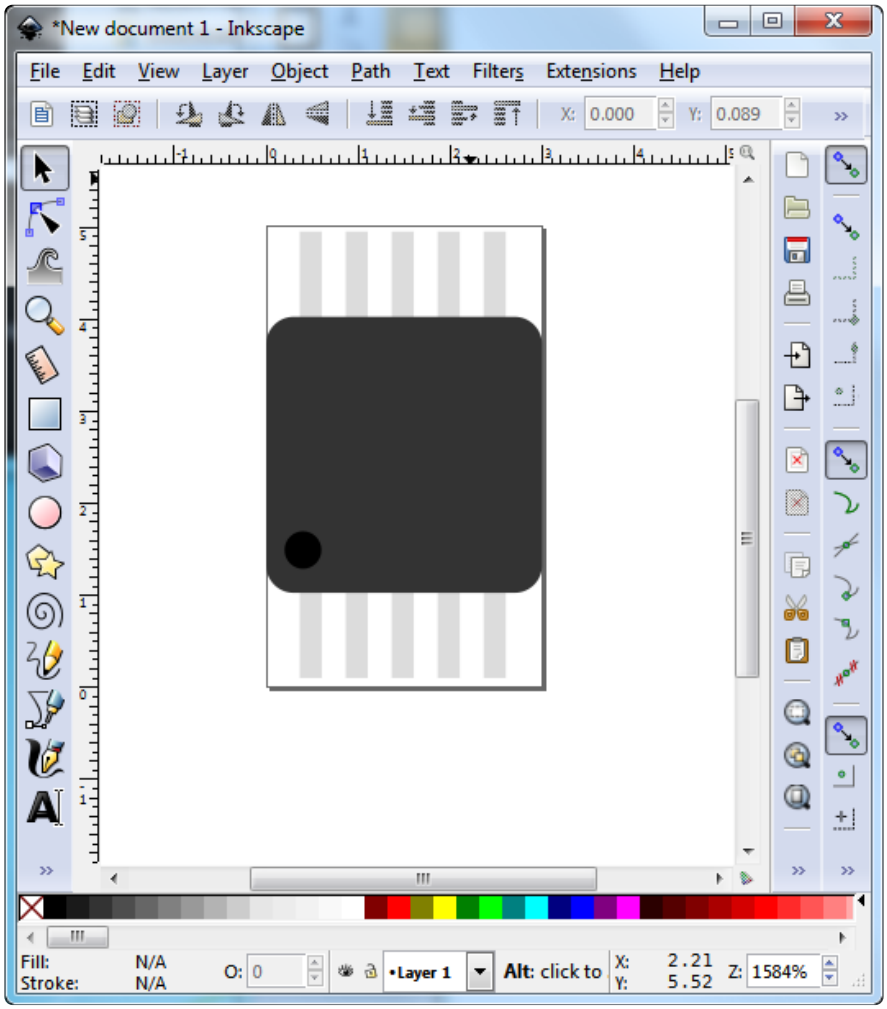
In this case we see the body is 3mm wide by about 5mm tall, set the size of the document to that size!



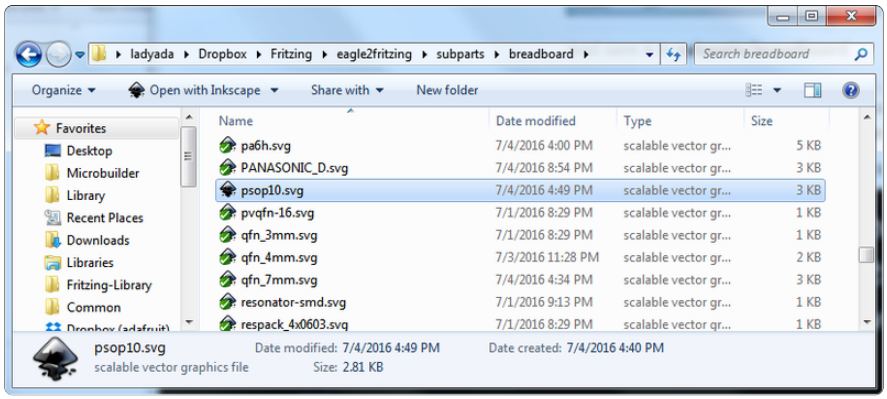
You can also open up the datasheet in Inkscape (forward it to the right page) and see if you can extract the diagram. This usually doesnt quite work but it can sometimes help to trace out the part



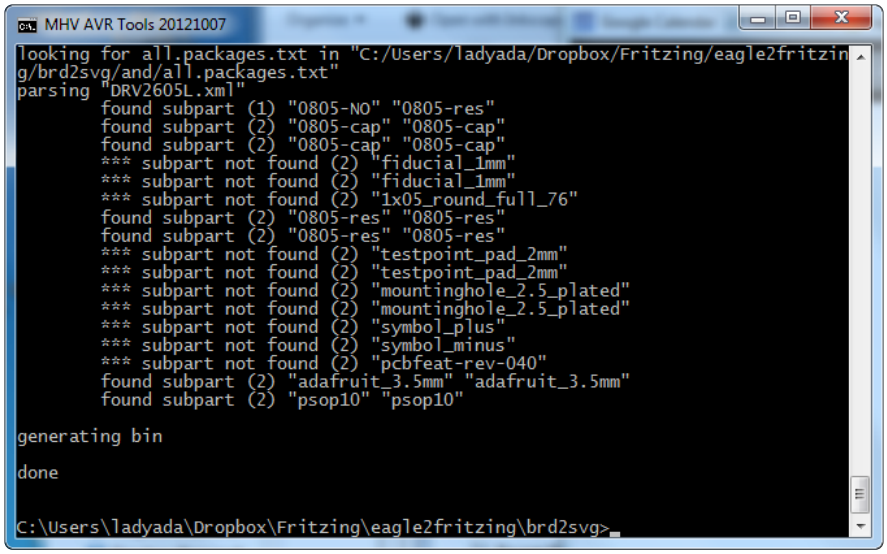
Anyways, do what you need to in order to draw the part in perfect scale!



And save it in subparts/breadboard



Rerun the tool, it will have found that part (if not check spelling and that you put it in the right location!)

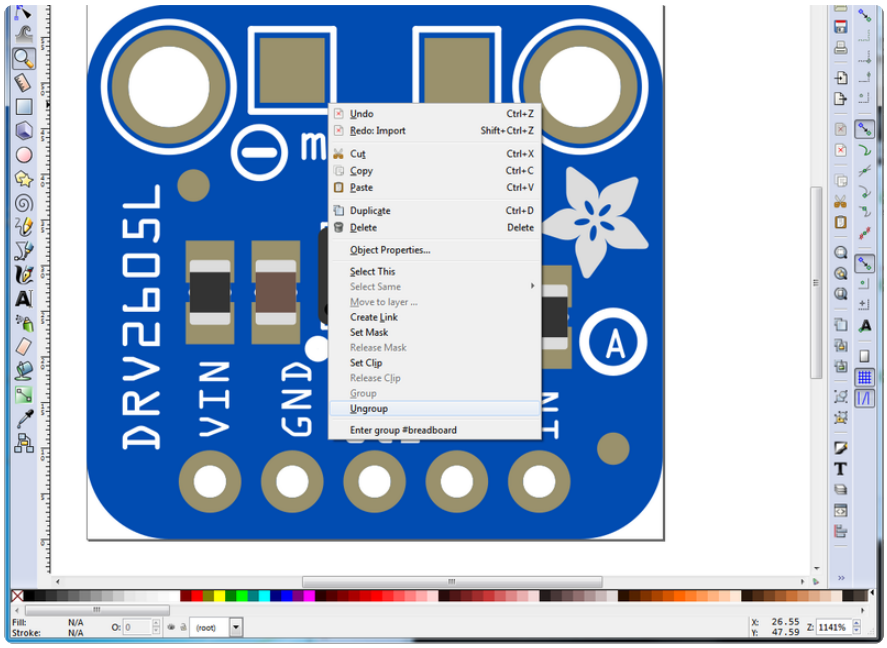


Moving Subparts

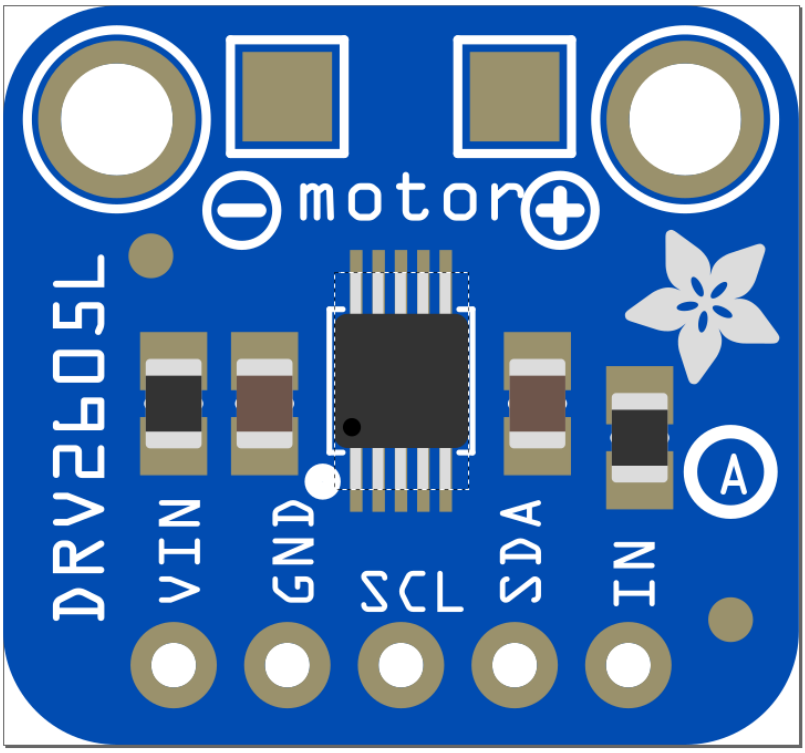
Often times the exact center isn't lined up. However it's very easy to fix.

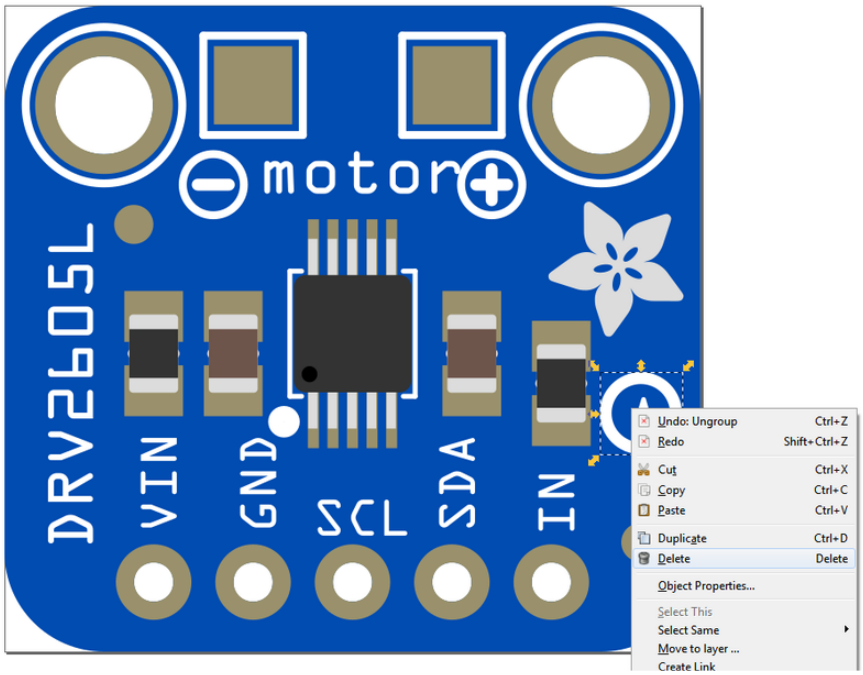
Do this after all subparts are found or alias'd since rerunning the tool will overwrite your breadboard.svg!

Ungroup the object



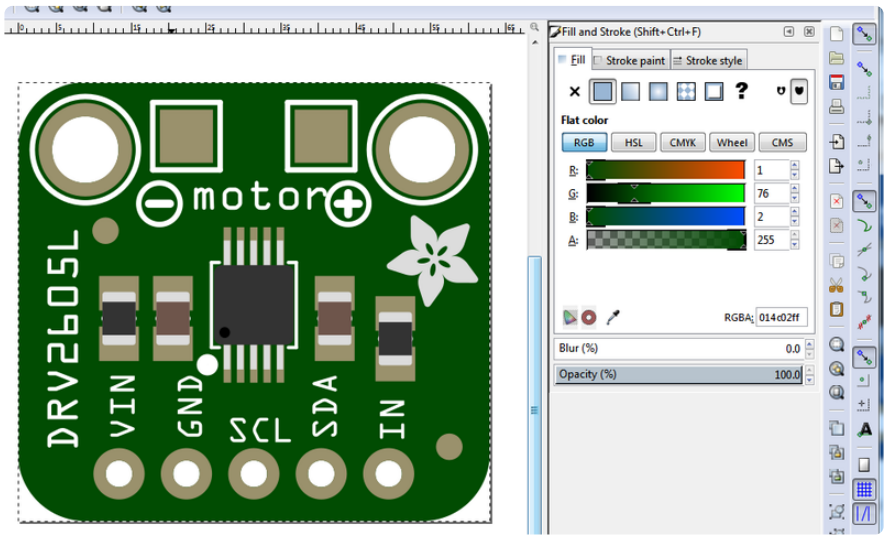
And move or remove anything you want





PCB Color

This is an easy one, but don't forget to change your PCB main polygon to the color you want



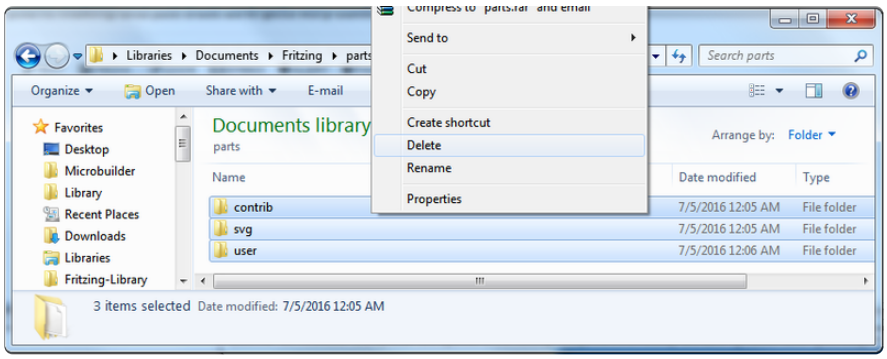
Save a backup once you have made your breadboard image the way you like! Rerunning the tool will regenerate the file and you'll lose all changes

Edit and Fix Fritzing File

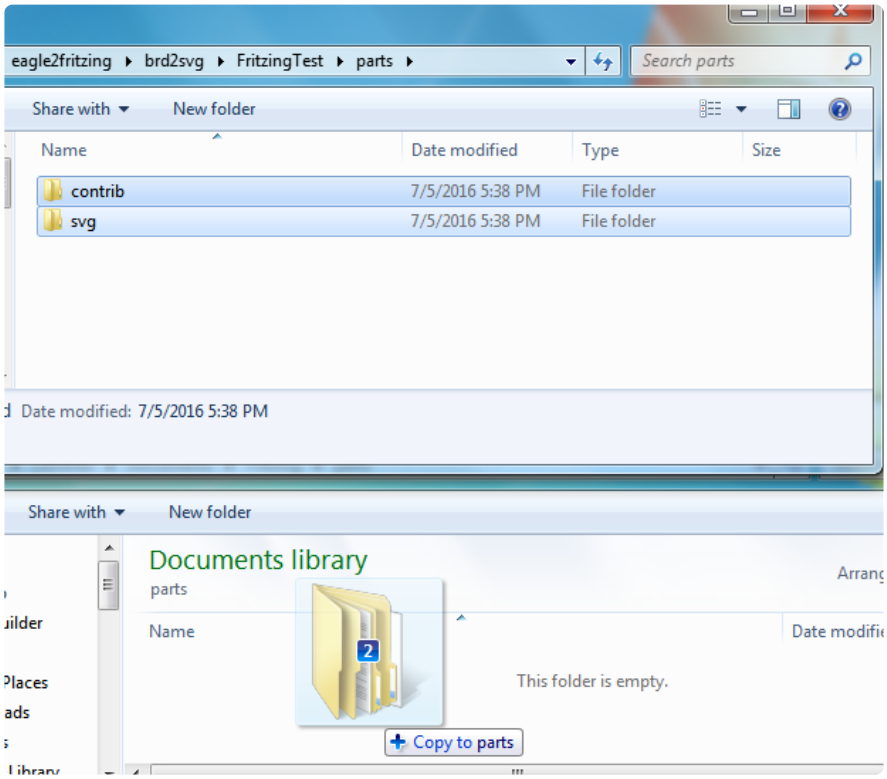
OK you are so close to done! Next up we will use the Fritzing Part Editor to really clean it up and fix everything.

In your Documents/Fritzing/parts folder is where the Fritzing App stores all your custom parts.

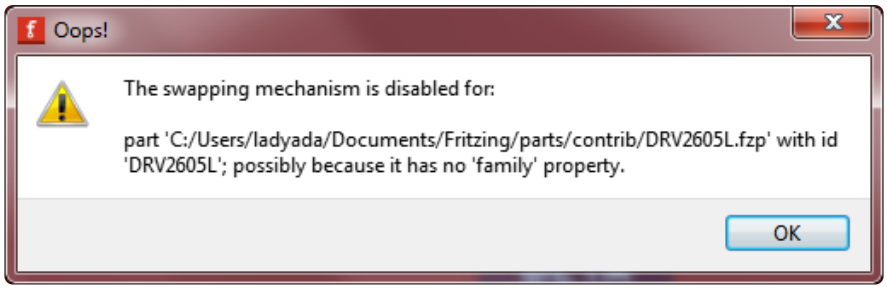
To make it easy and clean to handle revisions we suggest deleting everything in this folder (make a backup) since otherwise you may end up with multiple versions of the file you're making and just trust us it gets very confusing.



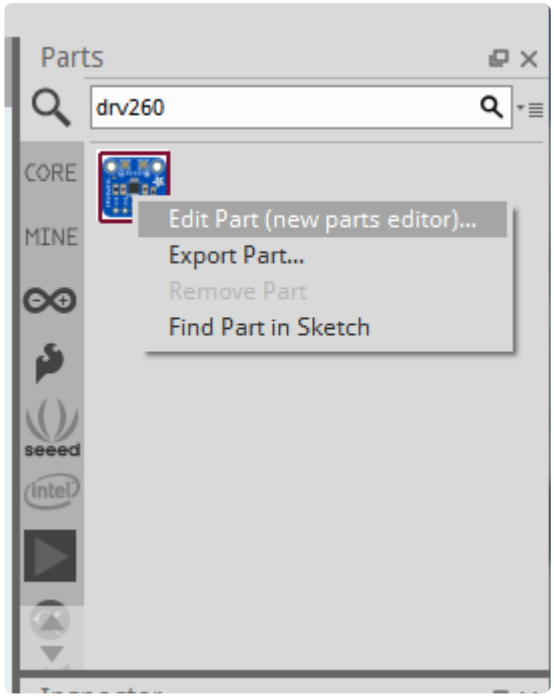
Copy over the contrib and svg folders from eagle2fritzing/brd2svg/FritzingTest/parts into your Documents/Fritzing/parts folder



Now start up Fritzing, you'll get a complaint about swapping, just ignore it.



Once open, in the Parts pane, search for the part and right click to edit

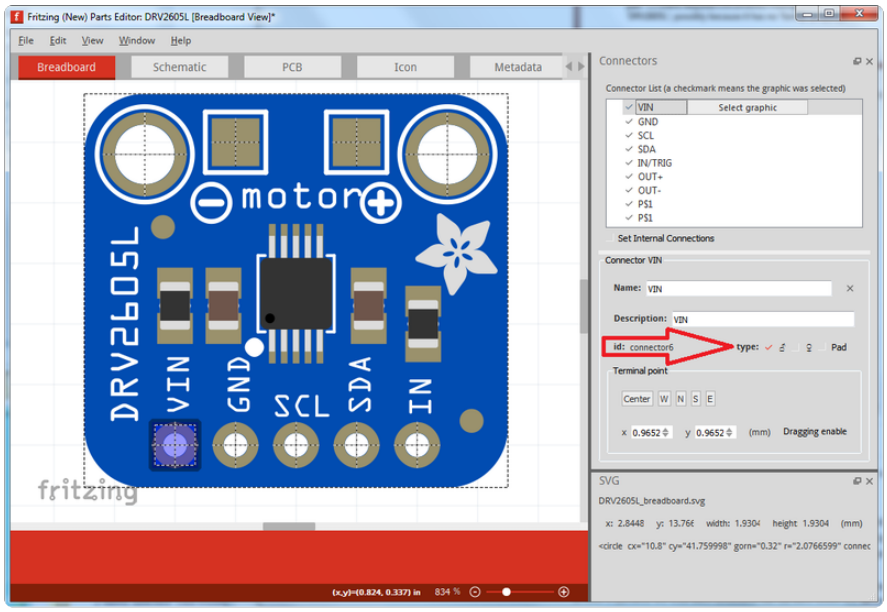


Breadboard Pin Gender

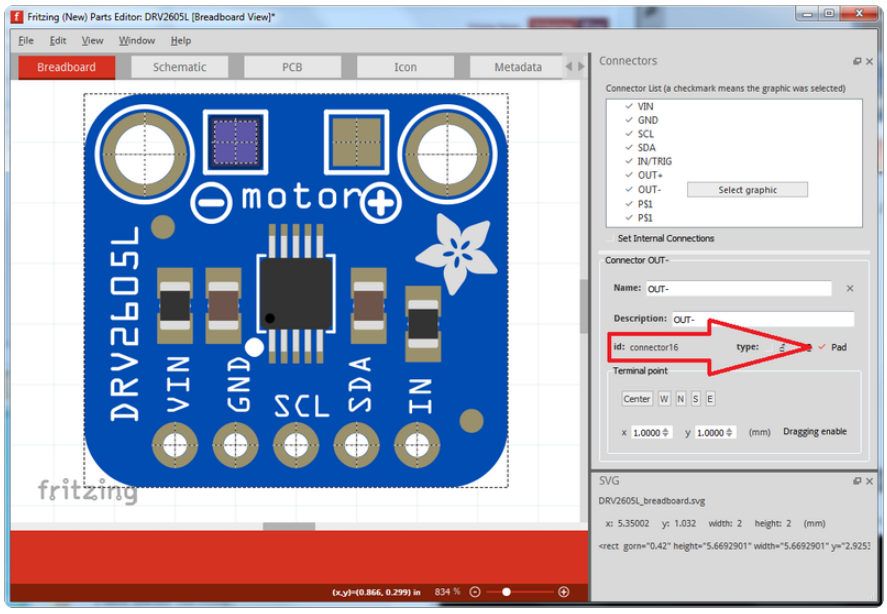
When you edit the part, the first tab is Breadboard view. It should look very familiar!

What's nice is each signal is already in the right hand pane and as you click on it you'll get information about that signal and a highlight of the SVG element that is the contact point

The contacts should all be in the right place. For breadboarding parts I prefer to set the gender of all the pins to Male - that way they will 'lock' into a solderless breadboard. Go through all the pads and click male for all breadboard pins



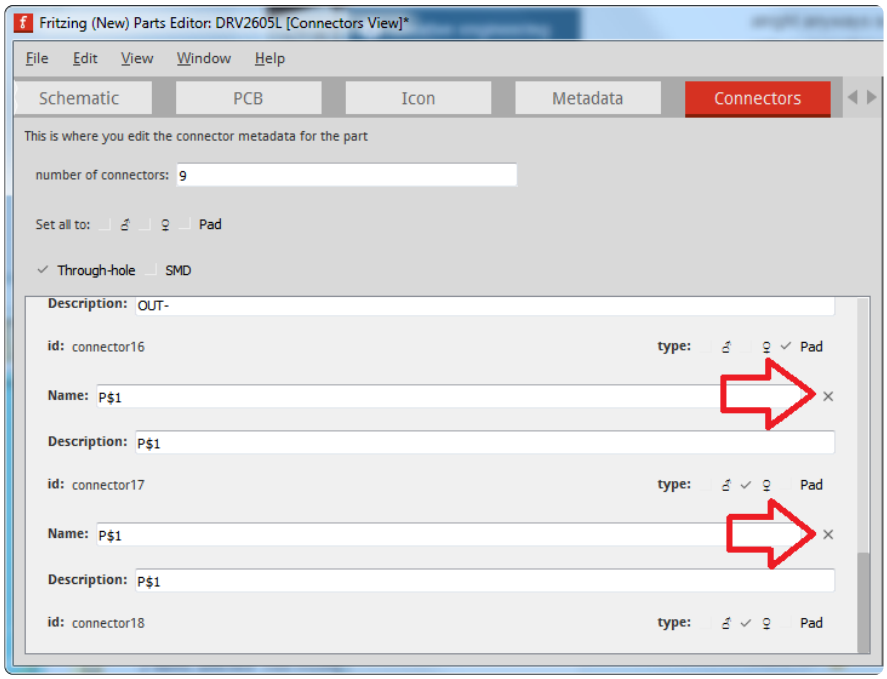
For the two SMT pads, you can set the connection type to Pad so it wont lock into a breadboard



There's also the extra mounting hole pads. They're not actually pads but its how we got them to appear in the breadboard view. To fix that...

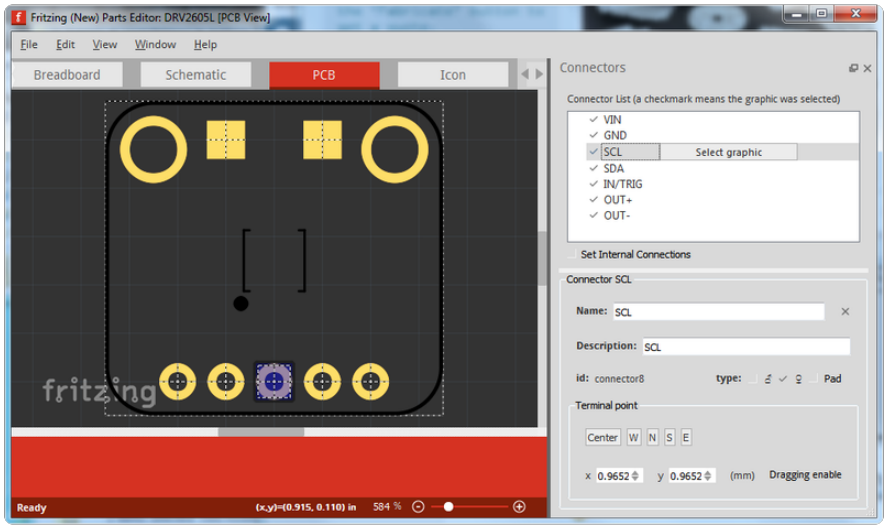
Removing Extra Connections

You can remove the extra connectors from within the Connectors tab. Scroll to the bottom and carefully remove the correct connections!



PCB View Review

You can now go through PCB view, check every connection and make sure they go to the right pad. If they're wrong use the Select Graphic button next to the connection in the Connector List to click on the SVG element that is the connection for that signal.

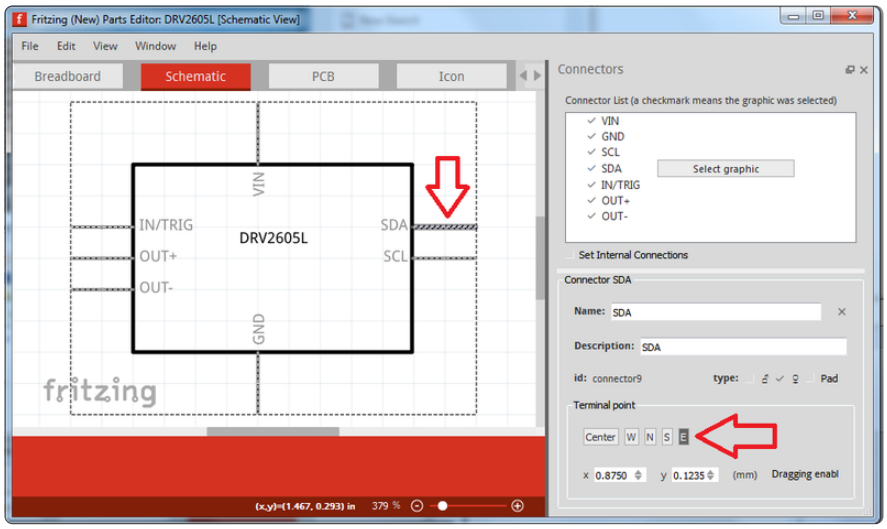


Note that the OUT- and OUT+ end up as pads on the PCB, not sure there's a way around this in order to keep the schematic and breadboard view connections. Fritzing isn't a hardcore CAD package, tradeoffs abound!

Schematic View Review

You can also review the Schematic view. Go through each connection in the schematic and verify its correct

One thing I've noticed if you've made hand edits to the schematic SVG (and even if you haven't) it's nice to select the terminal point for each connection

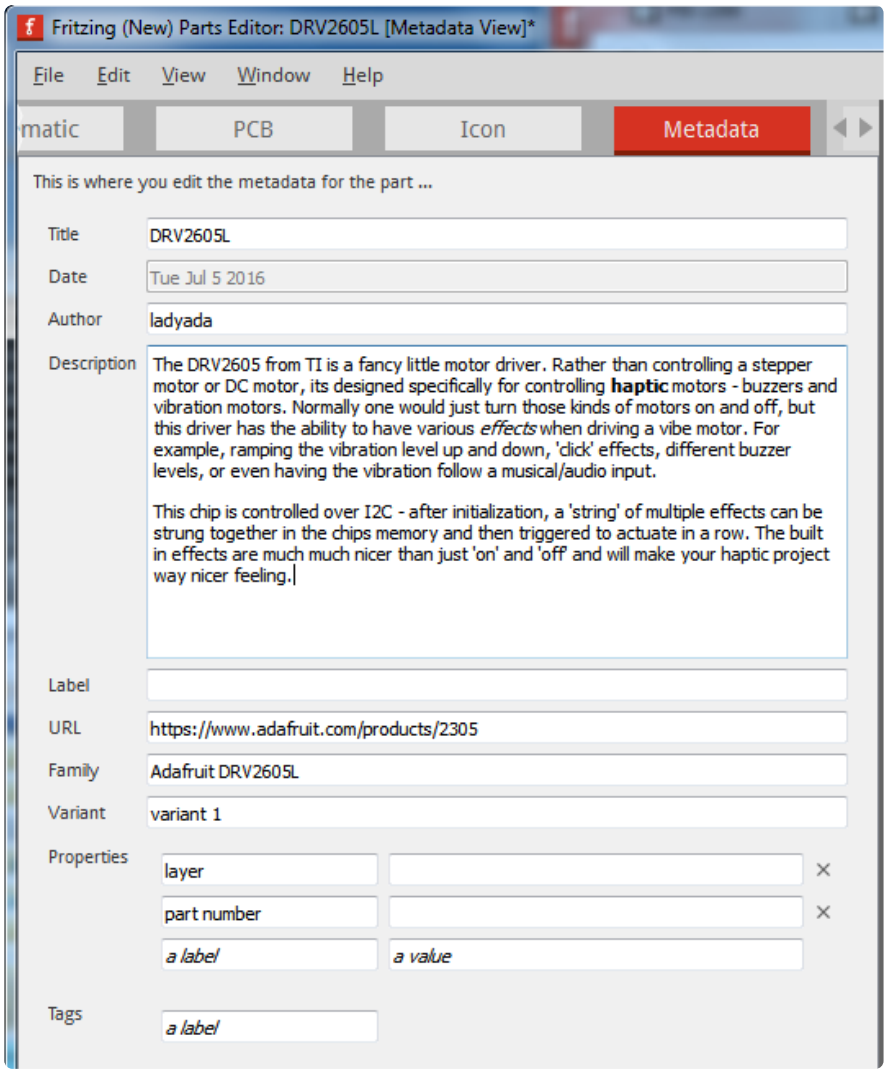


- For all the connections on the right side use East
- For all the connections on the left side use West
- For all the connections on the top side use North
- For all the connections on the bottom side use South

This will make wires that connect to the terminals come out of the end of the pin for a nice look!

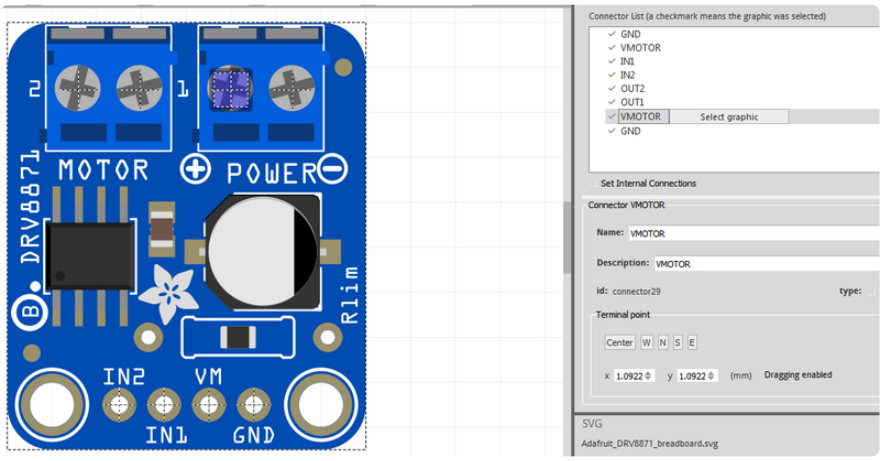
Metadata

Finally, put in the meta data for the part, including a description, URL, part number, etc!

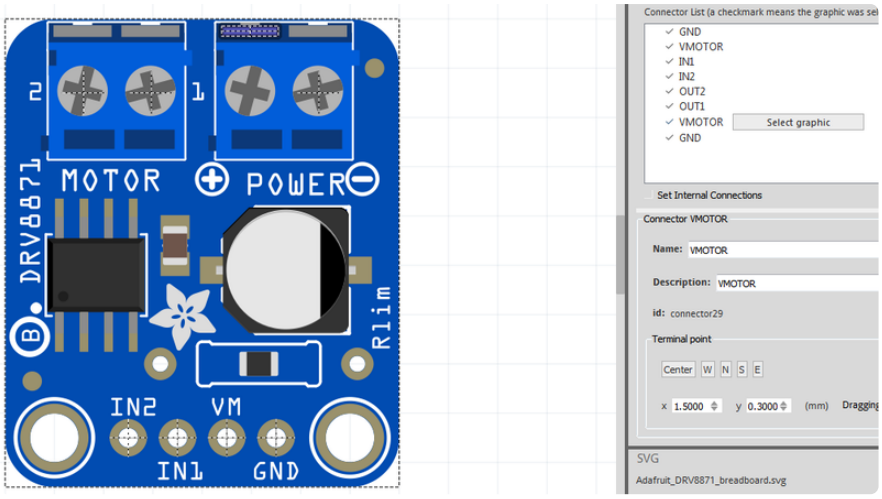


Terminal Blocks

Since we use terminal blocks a lot, this will come up from time to time. After adding the terminal block to the param file so they show up as a connection, you'll get them in the breadboard view but the connection point is the screw on top

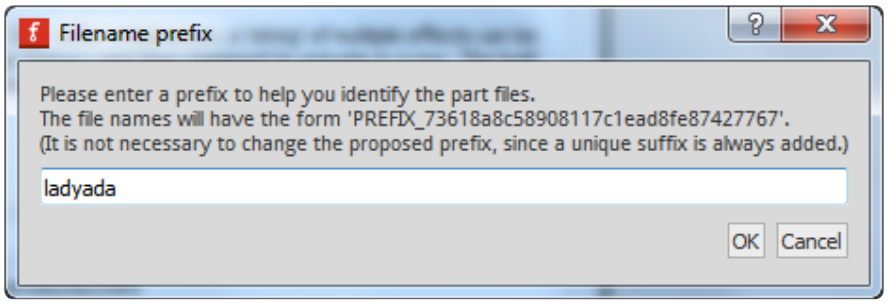


Use the Select graphic button to select the output port of the terminal block instead:

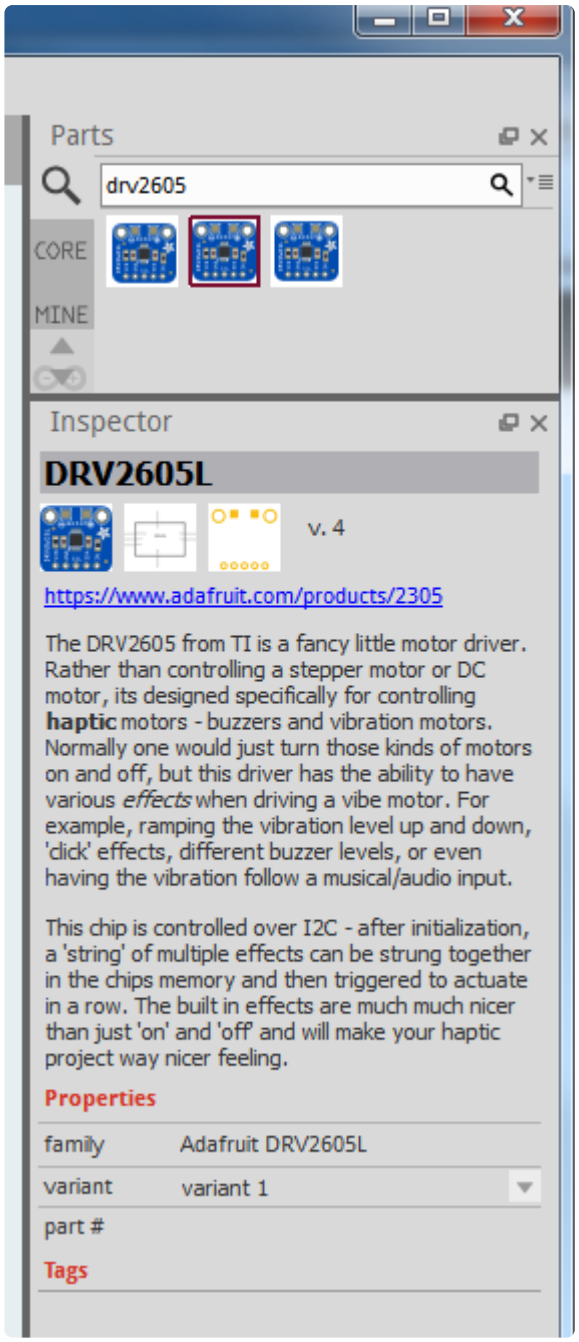


Test, Export, and Re-Edit Part

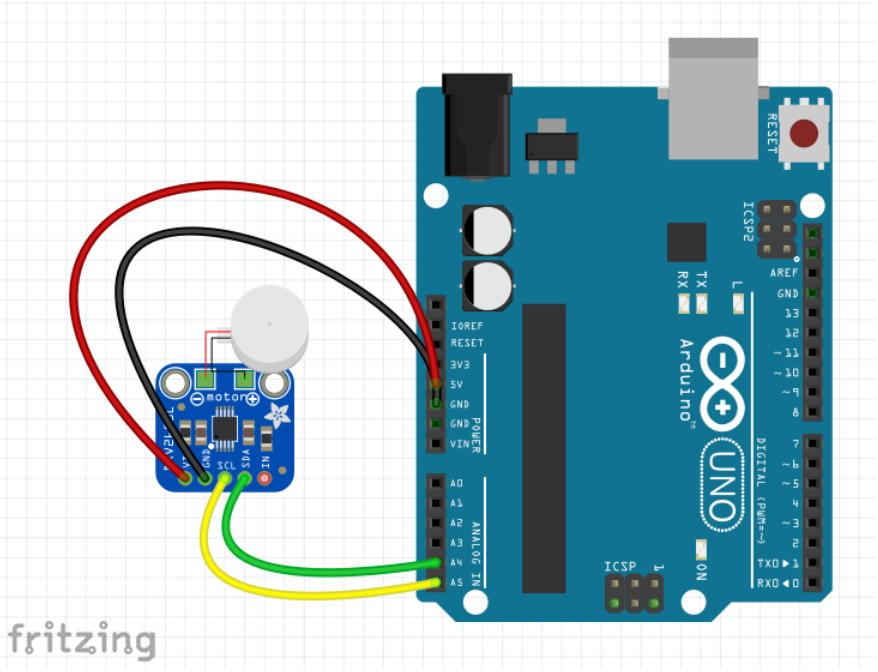
Finally, save your part and give it a prefix for parts that you've made (you can use anything to help you identify it)



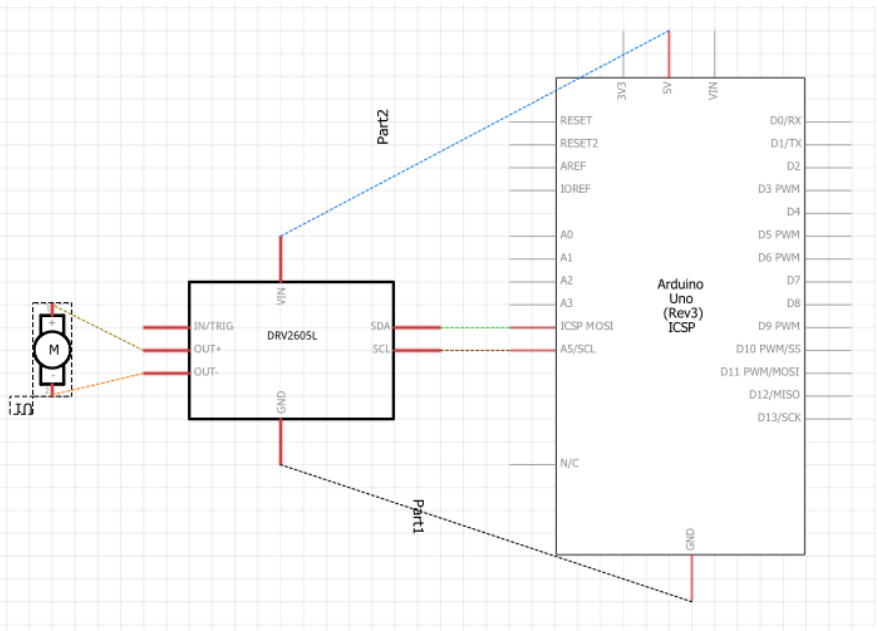
Close the parts editor and in the parts tab do another search to find your part again, you may have multiple versions so locate the final part you worked on



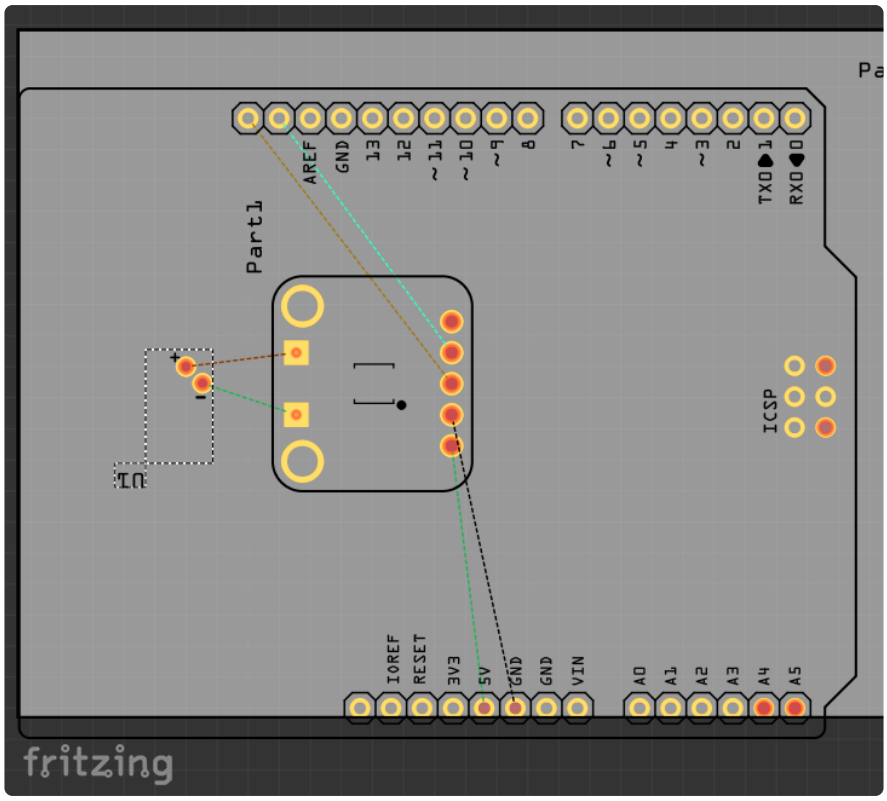
You can now try out your part, try connecting it to various other devices to make a circuit that uses all the pads



Be sure to also check out the schematic view (looks like the UNO has a weird icsp mistake but the DRV2605L is right!)



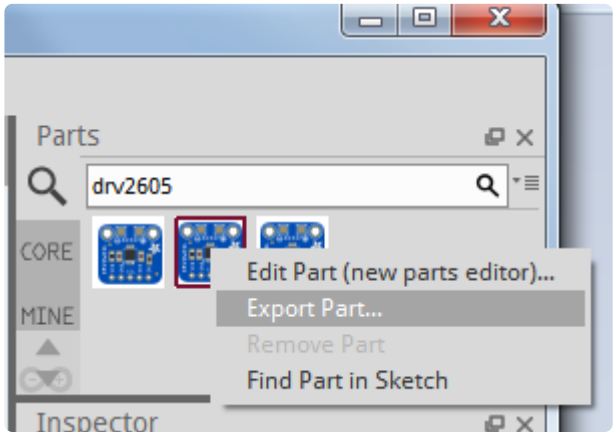
and the PCB view



You can still continue to edit the part, and do most anything that doesnt require editing a raw svg

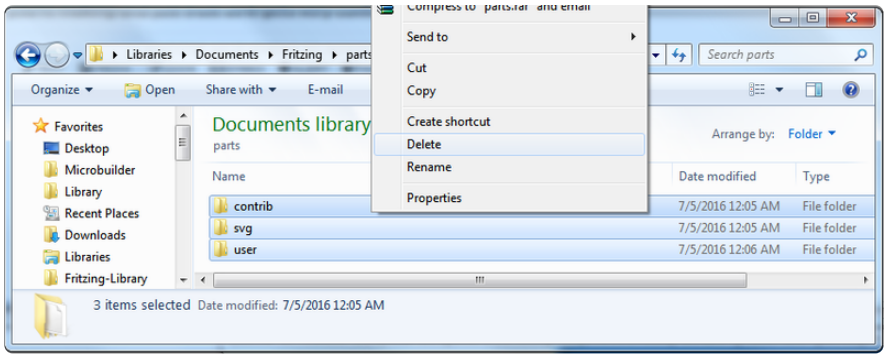
Export Fritzing Part

Right click to export it

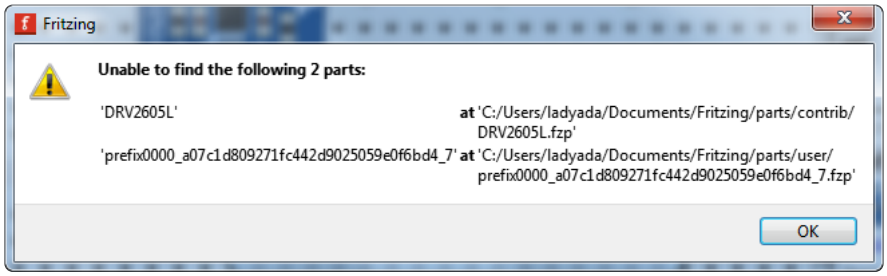


Save it somewhere safe!

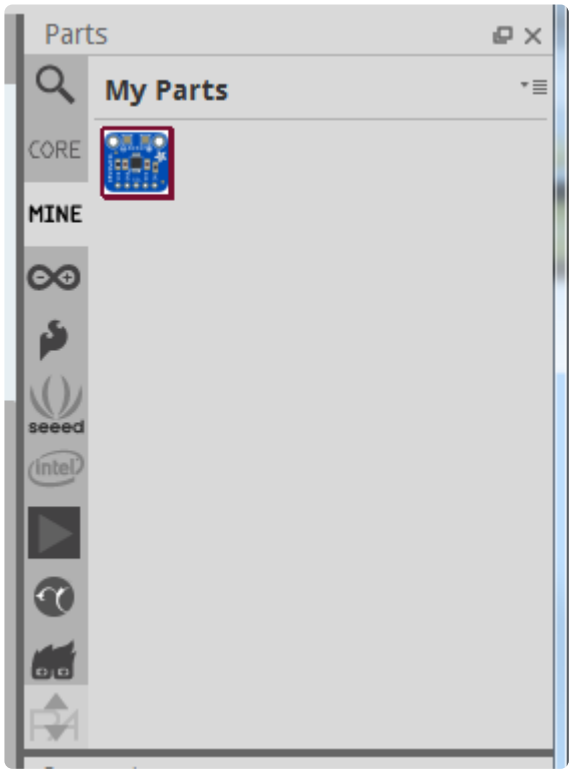
Now quit Fritzing and re-delete all the subfolders in your Documents/Fritzing/parts folder again



Now that you've cleared out all your partial edits, you can relaunch Fritzing. You'll get complaints that files are missing, that's OK! Just ignore them



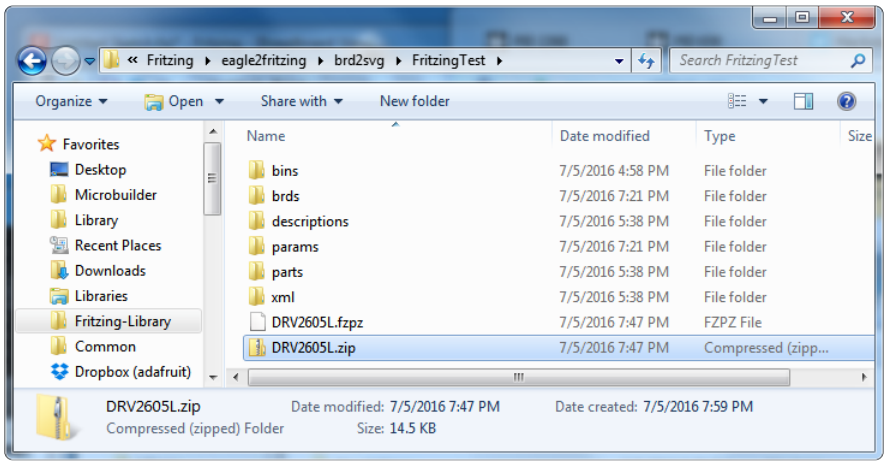
Open that fpzp file you saved earlier, the part will now appear in your My Parts bin!



Editing Post-Creation

OK so after some time you realize you want to tweak something with the part. You don't have to start over!

Export the part to a fzpz file, copy it and rename the copy with .zip instead of .fzpz



Inside, you'll find all those SVGs again! You can edit the svg carefully and place it back in the zip, then rename it to .fzpz again.

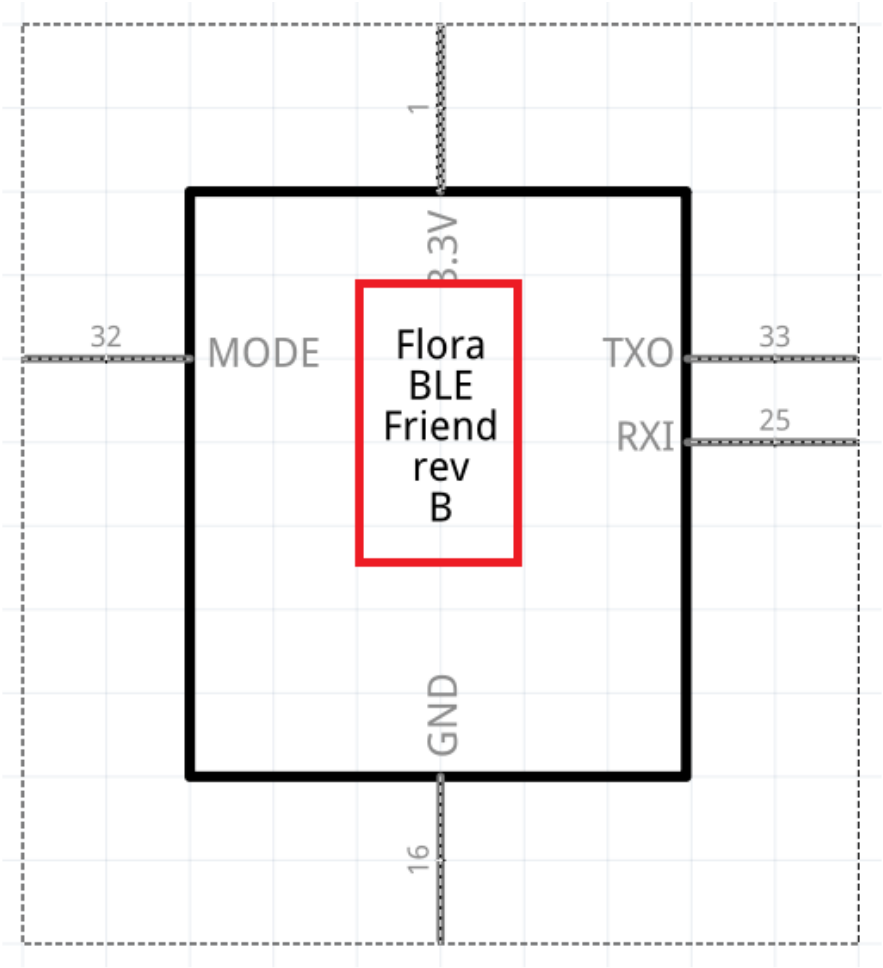
You may need to remove the part from your My Parts bin and re-import it. You may also need to delete all the parts from your Documents/Fritzing/parts folder since Fritzing throws a fit if you have two parts with the same name

---

Best Practices!

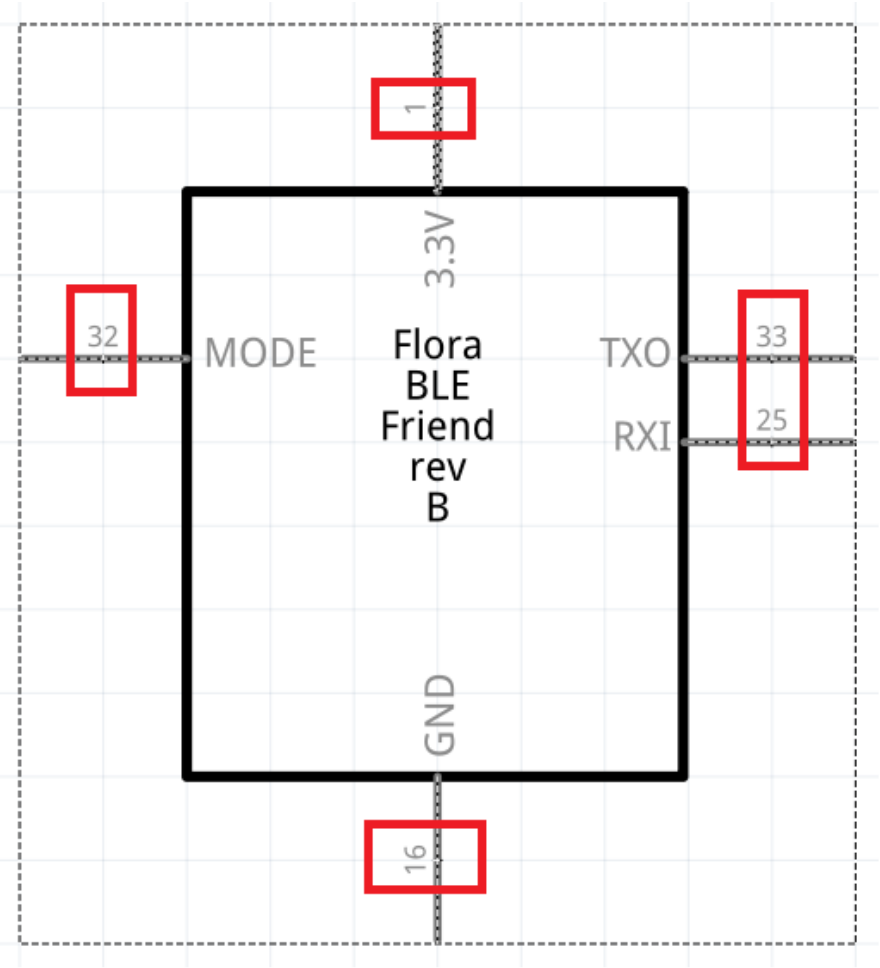
Rename file before conversion!

The text inside of the schematic view will match the file name, you can edit this later in Inkscape but its nicer to do it beforehand so rename it to something nice like 'Adafruit Flora Bluefruit' (no revision number, etc)



Remove pin numbers

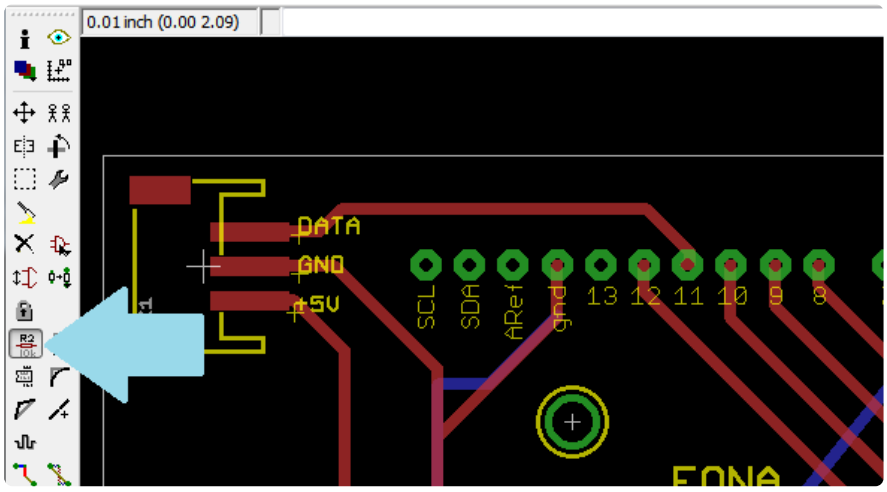
For some reason sometimes schematic view has pin/id numbers on the schematic. You can ungroup and just remove these

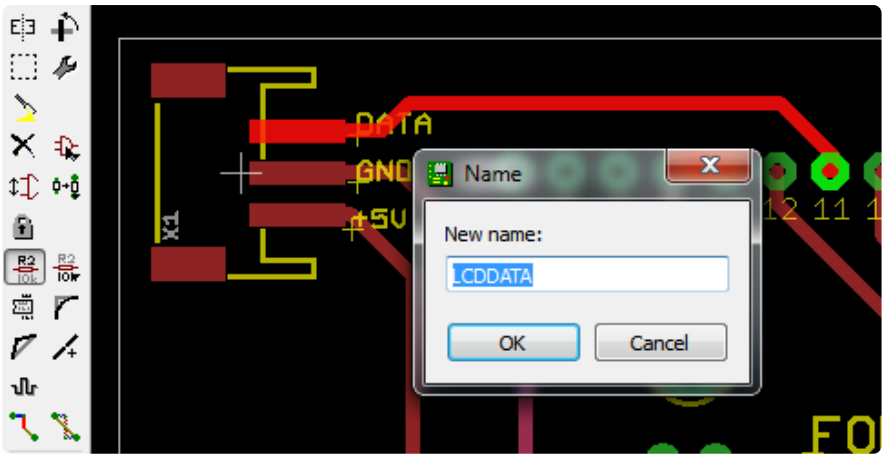


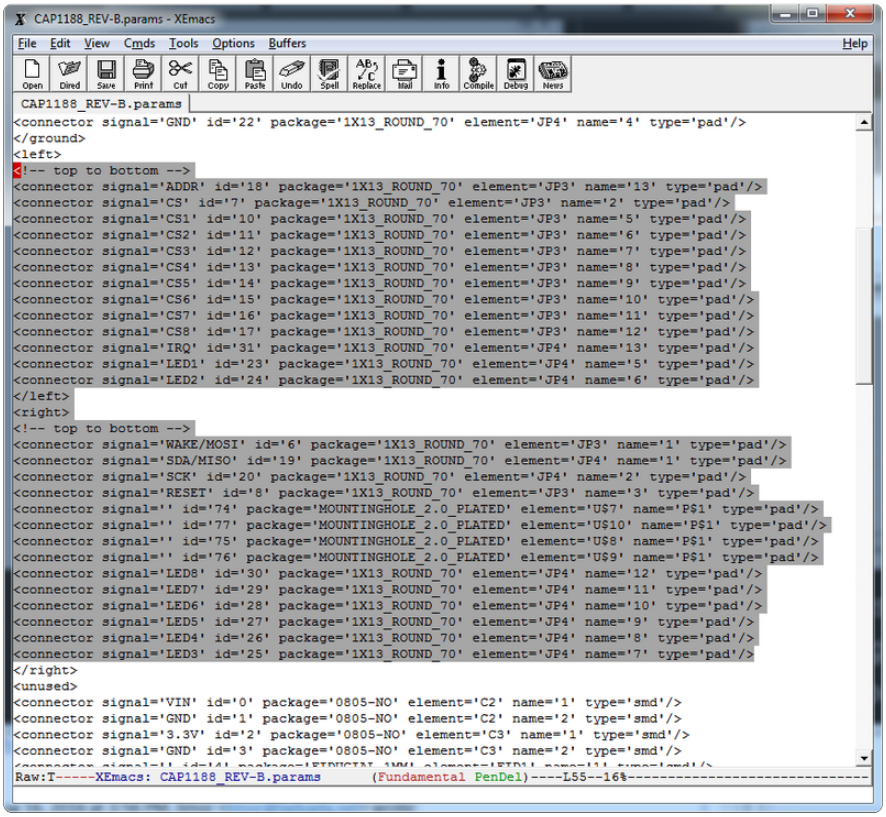
Rename Signals in Eagle

Instead of changing the names of signals in Inkscape/Fritzing you can just do it in EagleCAD

Ideally, rename signals so that they match what the silkscreen says





Rearrange pins in schematic

After you run the run.py generator once, you will create a params file in FritzingTest/params - its a text file, just open with any text editor. This is the file that is used to create the schematic and such. Once it is generated, you can edit it. This is good for rearranging pins in the schematic. Say if you want to make all the inputs on the left, and the outputs on the right, find the <left></left> and <right></right> section. Each line indicates a signal, you can copy and paste to move signals from side to side. You can also move pins up to the top (<power></power>) or bottom (<ground><ground>