marvis Documentation

Release 0.1

Joshua Murphy

Sep 28, 2018

Contents

1 User Documentation **1**

2 Developer Documentation **3**
- 2.1 Project Structure . . . 3
- 2.2 Installation . . . 4
- 2.3 Troubleshooting . . . 5

CHAPTER 1

User Documentation

CHAPTER 2

Developer Documentation

2.1 Project Structure

The documentation below is a redux of the official QGIS documentation, with additions or subtractions relevant to our plugin specifically.

2.1.1 QGIS Files

`__init.py__` Generally a staple of most Python projects, it is different with QGIS in that QGIS requires it to contrain a `classFactory()` function and optionally contains other intialization code. Generally speaking this is the entry point, and in Python this file is required in order to make your code importable as a module.

`metadata.txt` A file containing metadata that is of interest to QGIS, like project name, version, etc. Among other things this is what QGIS parses when we upload our plugin to the QGIS plugin repository. This metadata will be seen by users.

2.1.2 Marvis Files

`marvis.py` The main source code file of our Marvis plugin.

2.1.3 Qt Framework Files

`resources.qrc` An XML document generated by the Qt Framework's Qt Designer. Probably don't modify this by hand. It is translated into python as `resources.py` which also probably shouldn't be messed with manually.

`form.ui` Auto-generated description of the Qt GUI. Translated into Python as `form.py`

2.2 Installation

2.2.1 Requirements

- QGIS 3
- Python 3
- pip3
- Git
- PyQt5
- pb_tool
- virtualenv

2.2.2 (Optional) Compile QGIS 3.x

We are compiling QGIS from source because it's helpful to have access to the API code, but mainly because QGIS 3.x is not available in our package manager yet.

Refer to the QGIS docs if this overview is too succint.

```
# Important: install the dependencies they mention before proceeding.
# There are a large number of packages required to compile QGIS.

cd /usr/local/bin
sudo ln -s /usr/bin/ccache gcc
sudo ln -s /usr/bin/ccache g++

mkdir YOURDIR
cd YOURDIR && git clone https://github.com/qgis/qgis.it
cd QGIS
git fetch origin && git checkout release-3_2
mkdir build && cd build
ccmake .. # Hit the [c] key and the [g] at the menu to generate the defaults.
make -jX #Where X is your CPU core count
# It will be easier to run "make install" too.
./output/bin/qgis
```

2.2.3 Build Marvis

Ensuring you have `git` installed, on a command line, and in the directory you want to store the marvis code, run:

```
git clone https://github.com/ranguli/marvis
```

Then enter the marvis directory:

```
cd ./marvis
```

With no errors, move on to the next section.

Note: once you are ready to starting contributing to the code, you will need to switch to a different *branch.* This is outside the scope of the documentation.

2.2.4 Compilation

Compilation is perhaps a bit of a misnomer seen as how we're using an interpreted language, but there are certain files that do need to be collected or converted from another format into Python when one is developing plugins for QGIS. The equivalent of a "compiler" for us will be `pb_tool` which you should have installed with `pip` earlier.

To only **compile** the plugin, run:

```
pb_tool compile
```

To **compile and copy** the plugin to the QGIS plugins directory, run:

```
pb_tool deploy
```

Or, in shorthand:

```
pb_tool de
```

The tool is told exactly what to do using a config file, `pg_tool.cfg` in the project root directory. If you need to troubleshoot the behavior of `pb_tool` confirming that the config file is properly set up may be helpful.

The tool also comes with various other helpful features, such as `clean`

2.3 Troubleshooting

```
Error:  No module marvis.resources
```

In this case, you likely haven't run `pb_tool compile` in the root project directory to generate the `resources.py` file that is called as `marvis.resources`