MPHYG001 Greengraph Assignment

Simon Schaal, SN: 15072430

04.01.2016

1 Project layout

```
greengraph/
|-- CITATION.md
|-- greengraph
|   |-- command.py
|   |-- genGraph.py
|   |-- graph.py
|   |-- __init__.py
|   |-- map.py
|   `-- test
|       |-- fixtures
|       |   |-- london-green.png
|       |   |-- london-sat.png
|       |   |-- longlatplace.yaml
|       |   |-- longlatpng.yaml
|       |   |-- sea-green.png
|       |   |-- sea-sat.png
|       |   `-- startend.yaml
|       |-- __init__.py
|       |-- test_graph.py
|       `-- test_map.py
|-- LICENSE.md
|-- README.md
|-- report.pdf
|-- scripts
|   `-- greengraph
`-- setup.py
```

2 Command line entry point

```
usage: greengraph [-h] --from START --to END --steps STEPS [--out OUT]

Generates a graph of the proportion of green pixels in a series of satellite
images between two points

optional arguments:
  -h, --help     show this help message and exit
  --from START   Start city for Greengraph
```

```
--to END       End city for Greengraph
--steps STEPS  Number of images to process between the cities
--out OUT      Destination file to save the graph. Graph is displayed if no
               output specified.
```

3 Problems encountered during coursework

- `from` is a reserved keyword in python. I therefore needed to use the `dest=` attribute of argparse to be able to use `--from` as an argument.

- Using mocks in an appropriate way to realise testing wasn't always that easy and intuitive.

4 Advantages and costs involved in preparing a package for release

I think the benefits involved for preparing your work for release strongly overweigh the costs. The costs are especially low if one starts laying out an appropriate folder structure from the beginning for each project. If you've done this once than the setup.py, citation and license file can most likely be reused for the next project after a small amount of adjustments. Wrirting good tests for the project will consume a little more extra time. However, the tests are important to make sure the code works in the intended way and will help to find changes which broke the code much more easily. Writing and maintaining the tests while writing the code should minimise the extra cost.

The best way to have your code tested is by making other people use your code and thats the big advantage of preparing the code for release. It makes it very easy for others to install your code using `pip` for example. If uploaded on github others can even directly install from github. Furthermore, github offers a platform to report bugs and others can even push fixes and make suggestions which makes collaboration very easy, where the `git` version control allows to revert/merge changes at any time.

In summary, the goal of writing good software is to make people use it and to prevent others from writing their own software for solving the same problem. Ideally, the more people are using (/contributing to) the software the better the software should be.

5 Steps to build a community

- The first step would be to upload the project to github. This makes it available to anyone while offering others to contribute iby suggesting additional features, reporting bugs, pushing fixes, etc.
- After some testing project submission to the pypi project could be a next step, which would make the package available for a very large community.
- A final step would be to have the project submitted to package management systems such as `homebrew` for mac and `apt-get` for linux.